Simposio CEA de Robótica, Bioingeniería, Visión por Computador y Automática Marina 2026

Implementing kinematics in a rehabilitation CDPR: Patient-following mode

Riveiro, E.[a,*], Silva-Muñiz, D.[a], Pérez-García, U.[a], Garrido, J.[a]

[a] *EN.EDI Research group, Department of Automation and System Engineering, Universidade de Vigo, Maxwell 9, Vigo, 36310, Pontevedra Spain*

Resumen

El empleo de robots paralelos accionados por cables (CDPRs) como soporte a la rehabilitación de miembros inferiores es una opción atractiva dado que posibilitan amplios espacios de trabajo con restricciones mínimas de movilidad para el usuario. Este artículo presenta dos contribuciones. La primera consiste en la implementación de las cinemáticas de un CDPR de 8 cables en un controlador industrial, integrada como un módulo C++ en tiempo real. Apoyándose en esta, la segunda contribución desarrolla un modo de "seguimiento del paciente" guiado por unas gafas de realidad virtual que proporcionan referencias cartesianas para ser ejecutadas en el controlador industrial del CDPR. La propuesta se valida con dos experimentos: una evaluación sobre un prototipo a escala con trayectorias lineales y circulares, y un experimento con un participante realizando movimientos de rehabilitación de la marcha. Los resultados muestran un seguimiento preciso, con error RMS de posición 3D de 12.64 mm y latencia de 350 ms, demostrando viabilidad del modo propuesto.

Palabras clave: Arquitectura de software de control, Mecatrónica para sistemas robóticos, Planificación de tareas y movimientos, Realidad virtual, Robótica médica y de rehabilitación

Abstract

Cable-Driven Parallel Robots (CDPRs) are well suited for lower-limb rehabilitation as they provide large workspaces with minimal mobility constraints for the user. This paper makes two contributions. First, it presents the implementation of the direct and inverse kinematics of an 8-cable CDPR on an industrial controller, integrated as a real-time C++ TcCOM kinematic transformation. Second, it develops a VR-driven "patient-following" mode in which the headset provides Cartesian position references executed in the industrial controller. The approach is validated in two experimental studies: a controlled evaluation on a scaled prototype using repeatable linear and circular trajectories, and a full-scale test with a participant performing representative rehabilitation motions. Results show accurate and stable tracking, achieving a 3D RMS position error of 12.64 mm and an overall latency of approximately 350 ms. These results demonstrate the feasibility of integrating VR guidance, custom CDPR kinematics, and industrial motion control for patient-following rehabilitation scenarios.

Keywords: Control software architecture, Mechatronics for robotic systems, Medical and rehabilitation robotics, Task and motion planning, Virtual reality

1. Introduction

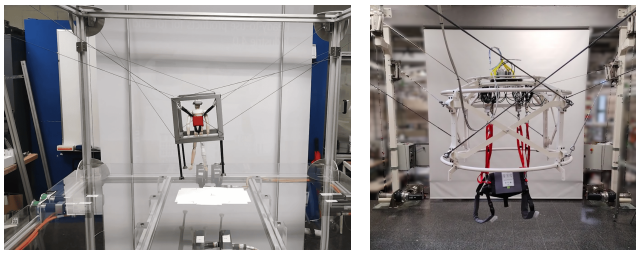
Medical rehabilitation aims to restore patients' motion abilities following injuries or chronic and degenerative conditions (Van der Loos and Reinkensmeyer, 2008). In lower-limb disabilities, gait impairments often lead to inefficient and unstable gait patterns and increase the risk of falls (Mikolajczyk et al., 2018). Rehabilitation technologies can promote transferable motor skills, intensify training, and enhance intervention effectiveness (Gonçalves et al., 2020). In particular, robotic gait rehabilitation can deliver precise, repetitive, and intensive training while reducing therapists' physical workload and allowing objective progress evaluation (Bruni et al., 2018). These systems have attracted interest in recent years due to the increasing need for rehabilitation treatments (Zuccon et al., 2024).

Among these technologies, cable-driven robotics stand out for their adaptability and for being less constraining to the patient than rigid-link solutions (Gonçalves et al., 2020; Oyman et al., 2021; Shoaib et al., 2021). Specifically, Cable-Driven Parallel Robots (CDPRs) are a type of cable-based robot that allow motion in 3D and coverage of large workspaces, enabling therapeutic exercises beyond predefined trajectories.

*Corresponding author: jgarri@uvigo.gal

These rehabilitation systems are increasingly supported by virtual reality (VR) to improve engagement and motivation (Luque-Moreno et al., 2015). Realistic interaction requires accurately coupling the user experience with the physical motion, which can be achieved through different sensing and tracking strategies (Li et al., 2019; Hamzeheinejad et al., 2021).

Within this context, the European project VirtualR3 (January 2024–March 2025) developed VR-supported rehabilitation environments based on CDPRs (Garrido et al., 2024), including both a small-scale prototype for testing and a full-scale CDPR (Figure 1). One of the operating modes developed in the project was the "patient-following" mode, in which the VR headset worn by the user acts as the master device providing the reference motion, while the CDPR acts as the slave system that tracks this motion and accompanies the patient during the exercise. This assistance enhances user confidence and safety, for example by providing a stable support point if the patient feels unstable, and is intended for the later stages of rehabilitation. The end-effector is designed with a surrounding railing that the patient may voluntarily use for reassurance or in case of emergency. In addition, the patient may wear a harness attached to the robot end-effector, but in "patient-following" mode it remains slack and provides support only in the event of a fall.



(a) (b)

Figura 1: CDPRs of VirtualR3 project. (a) Scaled prototype, (b) Full-scale.

While many industrial controllers provide ready-to-use kinematic transformations for widely adopted robot structures (e.g., serial 6-axis, gantry, or Delta-type systems), CDPRs are typically supported only in limited predefined forms (Garrido Campos et al., 2025). For instance, TwinCAT Kinematic Transformation includes cable kinematics with a small number of cables (e.g., 3D- and 4D-cable transformations), whereas a solution for the 8-cable configuration considered in this work is not readily available. Consequently, the inverse and direct kinematics must be developed and implemented specifically for the target industrial controller, using the extension mechanisms and software resources it provides (here, modular C++ components integrated into the control application).

To validate the proposed kinematic modules and control architecture, two experimental tests were carried out. First, in order to verify that trajectory tracking can be performed based on the information provided by the VR headset, the validation is carried out against the scaled prototype, executing two predefined trajectories in a systematical and controlled way. Second, after this validation, a full-scale experiment is performed on the full-scale CDPR, in which a person carries out a representative exercise in "patient-following" mode. This second study introduces additional variability and non-ideal effects associated with human motion, and demonstrates the applicability of the proposed approach beyond the controlled setup.

2. 8-cable CDPR kinematics

The kinematic model adopted for both CDPR prototypes is based on a vector loop-closure formulation. In this work, the cables were pretensioned and their elastic deformation was assumed to be negligible. Accordingly, they were modelled as massless and inextensible.

Figure 2 shows the loop-closure vector for the i-th cable of length l_i, with proximal A_i and distal B_i anchor points, defined with respect to the base frame O and the end-effector frame P, whose origin is located at the TCP (Tool Center Point).

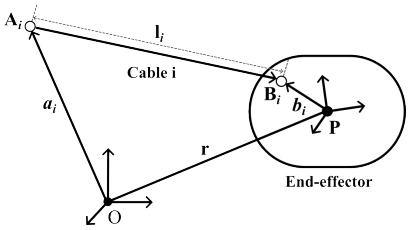


Figura 2: Vector loop-closure formulation for the i-th cable of the CDPR.

For the fully constrained CDPR with eight cables, (1) gives the inverse or backwards kinematics model:

$$l_i = \|\vec{L}_i\| = \|\vec{a}_i - R(\vec{b}_i) - \vec{r}\|, \quad i = 1, 2, \ldots, 8 \tag{1}$$

where, $\vec{r}$ denotes the desired position of the moving platform expressed in the base frame, $\vec{b}_i$ and $\vec{a}_i$ are the vectors from the TCP to the distal and proximal anchor points of the i-th cable, expressed in the end-effector and base frames, respectively, and R is the roll–pitch–yaw rotation matrix.

Direct or forward kinematics is posed as an optimization problem due to actuation redundancy (eight cables for six degrees of freedom) and unavoidable geometric and measurement uncertainties. The platform pose is estimated by minimizing the mismatch between the model-predicted cable lengths and the measured ones in a least-squares sense,

$$\min_{\mathbf{p}^* \in \mathbb{R}^6} \left\| \mathbf{l}(\mathbf{p}^*) - \mathbf{l}^* \right\|, \tag{2}$$

and solving the resulting nonlinear problem iteratively using the Gauss–Newton method (Bieber et al., 2023). The parameter to be optimised is the pose $\mathbf{p} \in \mathbb{R}^6$ of the end-effector.

3. Motion control architecture

This section presents the control architectures used to implement this operational mode. Specifically, Figure 3 illustrates the architecture in which the CDPR follows the real-time movements of the VR headset worn by the patient.

The industrial controller shown in Figure 3 uses a hierarchical structure of coordinate systems to translate VR-based motion into motor commands. The machine coordinate system (MCS, Figure 3), defined in PLCopen Motion Control Part 4 (PLCopen, 2008), serves as the primary reference frame, in which the VR headset position and velocity are expressed in Cartesian coordinates. These references are gotten by the user program (PLC task) and continuously updated with new position and velocity information, allowing the motion to be retriggered online. The resulting Cartesian command is then processed in the NC task (Numerical Control task), where setpoint

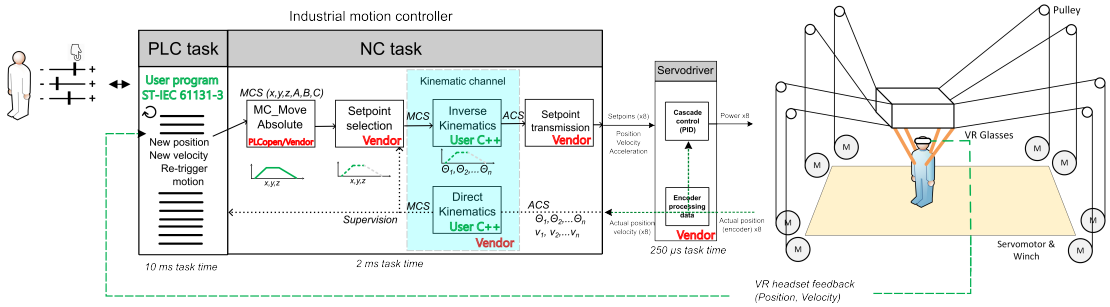


Figura 3: Architecture of the VR-guided CDPR mode. Integration of the physical CDPR system with the industrial control platform, including the PLC task, the NC kinematic channel, the user-developed ST program, the custom C++ direct and inverse kinematics, the servo-drive feedback and the VR headset signals.

selection is handled before entering the kinematic channel. The inverse kinematics transforms the trajectory from MCS to the axis coordinate system (ACS, Figure 3), determining the new cable references for synchronization of the eight axes. These ACS setpoints are subsequently sent to the servo drives, whose internal cascade control loops drive the winches and, through the pulley or eyelet routing, produce the desired motion of the end-effector.

In addition, Figure 3 shows the feedback and supervision path, in which encoder data from the servo drives are processed and used to reconstruct the Cartesian pose through the direct kinematics, which maps actuator-space information from ACS back to MCS. An accurate direct kinematic model is needed during the initialization of a kinematic group, i.e., a set of servo axes operating together under a specific kinematic transformation. When the group is created, the controller sends the calculated target position to move the servomotors. If the implemented direct kinematics does not match the real position of the servomotors, the discrepancy between calculated and actual values may generate incorrect initial setpoints, leading to high instantaneous accelerations and jerks in the servomotor shafts at start-up. Beyond this initialization stage, direct kinematics remains available at runtime for supervision purposes. In this work, it is specifically used for Cartesian tracking of the robot, enabling reconstruction of the actual platform motion so that the real trajectory can be monitored and compared with the VR-based reference.

3.1. Integration of CDPR kinematics using TwinCAT C++ and TcCOM modules

The 8-cable CDPR configuration considered in this work is not supported by the standard kinematic transformations available in the industrial motion controller. Therefore, the required kinematic mapping must be implemented as a custom transformation and integrated into the controller runtime. This could be achieved through real-time C++ modules encapsulated as TwinCAT Component Object Model (TcCOM) objects.

A TcCOM module is a real-time executable component that can be instantiated, configured, and scheduled within the TwinCAT runtime (Beckhoff Automation GmbH, 2025). Modules can be assigned to specific execution tasks (cycle time and priority), and expose input/output interfaces and parameters. These interfaces are described through a TwinCAT Module Class (TMC) file, enabling configuration and signal mapping in the same way as native motion, I/O, and control components.

Implementing the CDPR kinematics in C++ directly on the industrial controller provides several advantages over computation in the PLC task user program. One of them is that the kinematic transformation can be integrated directly with the motion control kernel, enabling seamless use of vendor-provided, validated resources for coordinated motion (e.g., axis groups, interpolators, and drive interfaces) (Beckhoff Automation GmbH, 2022). The alternative of computing the kinematics directly in the code of the PLC user program requires substantial additional development for trajectory generation, interpolation, and multi-axis coordination, thereby increasing engineering and validation effort (Silva et al., 2022).

3.2. TwinCAT-specific implementation of the 8-cable kinematic transformation.

The proposed C++ CDPR kinematics are deployed in a real-time C++ TcCOM module (CKinTrafoFB) that exposes the mandatory ITcNcTrafo interface.

The transformation maps the 8-axis actuator space (ACS, cable lengths) to the 6-axis Cartesian machine coordinate system (MCS), representing the platform pose $\mathbf{p} \in \mathbb{R}^6$ defined in Sec. 2. In TwinCAT, the interface ITcNcTrafo provides dedicated hooks to verify that the kinematic group and its inputs are valid: GetDimensions() specifies the expected sizes at activation (in this case, eight ACS axes and six MCS axes), and TrafoSupported() checks that the actual axis-group dimensions, configuration parameters, and incoming values are within acceptable limits before the transformation is executed. This prevents invalid group configurations and rejects physically infeasible commands.

At each controller cycle, the TcNcTrafoParameter structure provides the Cartesian reference (and, when available, its first and second derivatives). The module computes the eight cable-length setpoints by evaluating the loop-closure formulation introduced in Sec. 2. The resulting length references are written to the actuator outputs and, when coordinated motion requires derivative information, corresponding cable velocity and acceleration references are additionally provided using the available Cartesian derivatives and the associated Jacobian mapping. Geometric effects at the cable outlet (e.g., pulley routing) are compensated through specific configuration parameters (e.g., pulley diameter) that contribute to the effective commanded cable length.

Direct kinematics for monitoring are computed by solving the redundancy-induced least-squares problem posed in (2). The executed pose is reconstructed iteratively with a Gauss–Newton scheme: at each iteration, the residual between measured and model-predicted cable lengths is formed, an 8×6 Jacobian is assembled, and the pose update is obtained from the

normal equations $(J^\top J)\Delta\mathbf{p} = J^\top\mathbf{e}$ (Bieber et al., 2023). To ensure real-time compatibility, the developed C++ module uses a bounded iteration count and a configurable convergence threshold, and is initialized from the last commanded pose to improve convergence and robustness.

Robot geometry and configuration parameters (e.g., upper/lower frame and end-effector dimensions, end-effector rotations, pulley diameters, and default iteration settings) are exposed as module parameters through the TMC interface. Additional operational flags (e.g., TCP offset application) are provided via mapped process-data structures, enabling configuration and runtime diagnostics within the standard NC kinematic channel workflow.

The CDPR is configured as a kinematic channel comprising eight servo axes in the ACS associated with a Cartesian MCS, as illustrated in Figure 3. The proposed architecture combines two user-developed components: a PLC program written in Structured Text (ST) according to IEC 61131-3, and a custom C++ TcCOM kinematic transformation integrated into the NC task. The PLC program generates the high-level motion commands, whereas the TcCOM module implements the mandatory ITcNcTrafo interface and is instantiated as the user-defined NC transformation of the kinematic group. During interpolated motion, the NC kernel calls this transformation cyclically (task cycle of 2 ms) to map Cartesian setpoints expressed in MCS to actuator-space cable-length setpoints in ACS via Backward(), thereby enabling coordinated motion of the 8-axis group in Cartesian space. The complementary Forward() method reconstructs the Cartesian pose from the measured axis values for supervision and feedback. As shown in Figure 3, these user-developed functions are embedded between the vendor-specific setpoint selection and setpoint transmission stages, while low-level servo-drive control remains part of the manufacturer's hardware. Finally, motion commands are issued through standard PLCopen Motion Control function blocks acting on the kinematic group (Figure 3), whereas the NC-to-PLC channel provides coordinated-motion feedback and status in compliance with the PLCopen workflow.

4. Development of "patient-following" operation mode

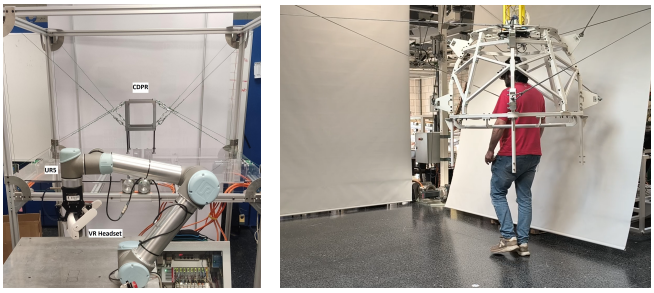
In the patient-following mode, the robot follows the movement of a person using only the position and velocity from a VR headset, with no physical contact between the person and the robot.

This mode was implemented by issuing MC_MoveAbsolute function block commands to the Cartesian axes in the MCS. Each VR headset sample is treated as an updated Cartesian absolute setpoint in $\{X, Y, Z\}$. By default, the MC_MoveAbsolute function block stops the motion once the commanded target position is reached. Therefore, to achieve continuous tracking, the developed user program updates the target and retriggers the function block through the edge-triggered Execute input whenever a new VR sample is received, issuing new setpoints before the previous segment is completed. This command-chaining logic prevents stop–go behavior. Safety is ensured by speed and acceleration limits at both the motion-command level and the axis-configuration level.

4.1. Evaluation with scaled prototype

The following experiment has been carried out to evaluate whether a VR headset can be a reliable "master" for a CDPR robot using the scaled prototype before deploying it in the full-scale one.

The experimental setup was designed as follows. The system integrated a Meta Quest 3 VR headset, a UR5 collaborative robot, and an 8-cable CDPR, as shown in Figure 4(a). The VR headset was mounted on a collaborative robot's end effector. The VR headset captured the real-time displacement, which was transmitted via TCP (Wi-Fi) to the CDPR motion controller at intervals of approximately 60 ms. The CDPR then attempted to follow the movement as closely as possible in both trajectory and timing.



(a) (b)

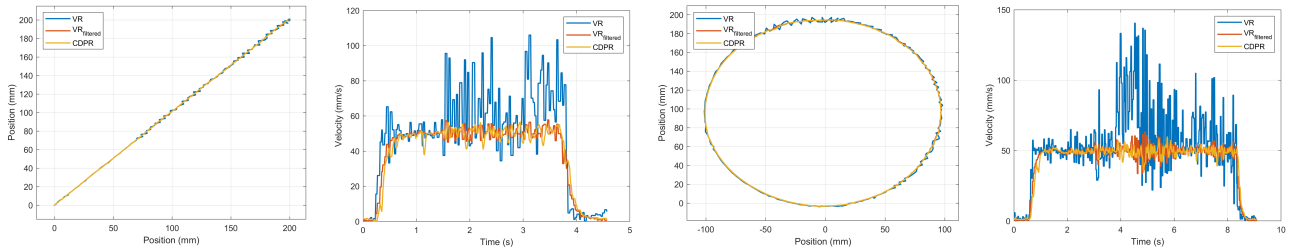
Figura 4: Experimental setup for the CDPR. (a) Scaled prototype, (b) Full-scale.

The UR5 robot was used to ensure high repeatability, thus eliminating the variability inherent in performing this movement with a person wearing the VR headset. This robot performed two specific trajectories: diagonal and circular displacements along both the X and Y axes. Each trajectory was performed five times to ensure result reproducibility and robustness. Also, both trajectories were carried out at a speed of 50 and 200 mm/s (according to gait speed of rehabilitation patients in (Barthuly et al., 2012)).

To reduce noise in the VR headset signals, a Kalman filter was also applied to the motion data to further command the CDPR. Both filtered and unfiltered data were tested under each condition. Four configurations were tested: unfiltered signals, position filtering only, velocity filtering only, and simultaneous position and velocity filtering. The Kalman filter parameters were Q = 0.1 and R = 1 for both position and velocity signals, chosen after several tests.

Figure 5 reports the line (200 mm in X and Y) and circle (100 mm radius) experiments, showing raw and Kalman-filtered reference signals (including delay) together with the CDPR position and velocity following. Table 1 summarizes the quantitative results of both experiments. The RMS positional error measures the average deviation from the ideal trajectory, the RMS velocity error quantifies differences in movement magnitude relative to the reference, and the Fréchet distance evaluates the overall similarity between trajectories while accounting for continuity and shape.

Table 1 and Figure 5 show accurate trajectory tracking, while the raw VR signal mainly degrades velocity. For the line segment at 50 mm/s, the unfiltered case yields 1,874 ± 0,073 mm RMS position error but large RMS velocity error



(a) (b) (c) (d)

Figura 5: Results for line trajectory execution at 50 mm/s. MC_MoveAbsolute: (a) Position following. (b) Velocity following. Results for circular trajectory execution at 50 mm/s. MC_MoveAbsolute: (c) Position following. (d) Velocity following.

Tabla 1: Comparison of metric errors for linear and circular trajectories at 50 and 200 mm/s using MC_MoveAbsolute with different filtering options.

Path	Filter	RMS pos error (mm)		RMS vel error (mm/s)		Fréchet distance	
		50 mm/s	200 mm/s	50 mm/s	200 mm/s	50 mm/s	200 mm/s
Line	-	1,874 ± 0,073	8,187 ± 1,607	27,594 ± 17,978	16,261 ± 5,820	3,241 ± 1,387	5,758 ± 0,367
	Pos	1,375 ± 0,165	8,029 ± 0,379	41,994 ± 2,362	19,891 ± 3,144	2,147 ± 1,128	5,876 ± 0,490
	Vel	2,716 ± 0,152	7,183 ± 0,593	7,844 ± 1,848	17,515 ± 2,669	2,743 ± 0,475	6,668 ± 0,706
	Both	1,050 ± 0,031	6,757 ± 1,033	5,775 ± 2,208	15,107 ± 6,725	1,672 ± 0,165	5,515 ± 0,141
Circle	-	2,814 ± 0,179	11,409 ± 0,522	42,455 ± 8,876	40,200 ± 3,4961	4,732 ± 0,468	7,508 ± 0,674
	Pos	1,999 ± 0,406	10,420 ± 0,887	41,482 ± 10,305	44,835 ± 4,650	2,509 ± 1,345	6,840 ± 0,865
	Vel	3,869 ± 0,591	9,770 ± 1,043	9,927 ± 0,704	31,530 ± 10,593	6,027 ± 1,246	8,786 ± 3,160
	Both	1,198 ± 0,243	9,202 ± 1,156	6,689 ± 2,740	32,491 ± 4,409	1,815 ± 0,321	6,894 ± 1,117

(27,594 ± 17,978 mm/s). Kalman filtering of both position and velocity provides the best overall results, reducing errors to 1,050 ± 0,031 mm and 5,775 ± 2,208 mm/s and improving trajectory similarity (Fréchet distance 1,672 ± 0,165 mm). At 200 mm/s, the same configuration remains the most consistent for the line trajectory, with 6,757 ± 1,033 mm RMS position error and 5,515 ± 0,141 mm Fréchet distance.

For the circle, filtering mainly reduces jitter and improves velocity smoothness, while path similarity is comparable across cases. Kalman filtering added about 60 ms of delay to TCP socket communication, and the average delay from sending the command to the robot reaching a similar point was about 140 ms, resulting in a total latency of approximately 260 ms. Overall, the results support stable and accurate "patient-following" when filtering is applied.

5. Evaluation under real environment (full-scale CDPR)

An additional experimental study was carried out to assess the system performance in a more realistic rehabilitation scenario involving a human participant and the full-scale industrial 8-cable CDPR rehabilitation system shown in Figure 4(b). The objective of this evaluation was to verify that the proposed "patient-following" mode can reliably accompany human motion under natural gait variability.

5.1. Experiment

The experiment was conducted with one healthy participant under prior ethical approval and informed consent. From a known initial pose, the participant followed a straight segment, then a circular arc, and continued along a second straight segment before returning to the starting point, while viewing an engaging virtual scenario through the VR headset.

Prior to starting the exercise, passthrough mode was used. A one-time alignment procedure was performed by the operator when launching the VR application by sequentially defining three reference points using a hand controller: one for the initial pose and two defining the orientation of the X-Y axes. After initialization, the reference frames of the VR application and the robot controller remained aligned.

The industrial controller (C-6030) executed the coordinated motion using the same kinematic transformation and control strategy described in Sec. 2 and Sec. 3. The PLC task ran with a cycle time of 10 ms, and the motion and kinematics control task ran at 2 ms. To ensure safe operation, motion constraints were applied at the controller level, including limits on maximum velocity and acceleration. The maximum achievable end-effector velocity was limited to 400 mm/s due to actuator and gearbox specifications and a maximum permissible acceleration of 1500 mm/s^2.

5.2. Results and discussion

Figure 6 compares three trajectories: the reference trajectory obtained from the VR headset, the corresponding Kalman-filtered trajectory used for command generation, and the executed Cartesian trajectory obtained from the direct kinematics.

Robot tracking of the VR headset was performed in the X, Y, Z coordinates, yielding an average RMS 3D position error of 12.64 mm and a Fréchet distance of 29.34 mm. The end-to-end tracking delay, estimated by cross-correlation between the commanded and executed trajectories, was 230 ms. This value should be interpreted as an effective tracking delay rather than the time required to reach exactly the same discrete point, since the position setpoints were updated cyclically at each PLC cycle to avoid motion interruption when a target was reached, as described in Section 4. When the additional delays introduced by the VR headset and the filtering stage are included (approximately 60 ms each), the total latency from the headset motion to the robot response is approximately 350 ms.

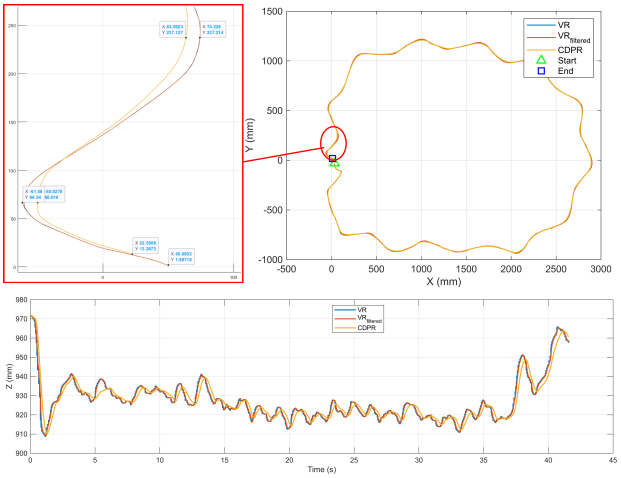


Figura 6: 3D trajectory carried out in the experiment. The top panel shows the X–Y trajectory, including the initial and final points and a detailed view of the largest error between commanded and actual values. The bottom panel shows the Z trajectory over time.

These results confirm that the proposed "patient-following" mode allows the full-scale CDPR to accompany the user during rehabilitation motions. Across the experiment, the robot reproduced the path suitably during the walking and curved segments while remaining synchronized with the user. The main limitations were VR headset noise, communication, and filtering latency. Moreover, robot acceleration had to be limited to avoid sudden movements that could result in unsafe pushes to users. Nevertheless, the measured trajectory deviation remained below 30 mm (Fréchet distance and RMS error), the delay was compatible with a smooth user experience, and no unstable behavior or discontinuities were observed. These findings support the applicability of this opearion mode beyond the validation setup, showing that the industrial control architecture and custom kinematic transformation can be deployed on a rehabilitation-grade CDPR in human-in-the-loop scenarios.

6. Conclusions

An 8-cable CDPR kinematic transformation (inverse and direct kinematics) was implemented as a real-time TcCOM module and integrated into an industrial motion controller, enabling Cartesian command execution through standard PLCopen motion resources. Building on this implementation, a "patient-following" mode was demonstrated, where the VR headset data provides the reference for motion generation.

Experimental results on both a controlled setup with a scaled prototype and the full-scale rehabilitation CDPR show that the approach enables stable accompaniment of user motion, and that Kalman filtering improves command smoothness and tracking consistency. The use of "proven industrial devices", such as the industrial motion controller, also paves the way for future medical certification of the developed CDPR system.

Future work will extend the method to stair ascent/descent tasks, squads, incorporating data from complementary sensors (e.g., IMUs or vision cameras) to compensate angular deviations and improve tracking accuracy in complex movements.

Acknowledgements

This research was partly funded by the VirtualR3 project, as part of the EMIL project (HORIZON-CL4-2021-HUMAN-01-06, EMIL Grant Agreement No 101070533) and by the Xunta de Galicia under project Grupo EN.EDI GPC-ED431B 2025/41 (Grupos de Potencial Crecimiento).

References

Barthuly, A. M., Bohannon, R. W., Gorack, W., May 2012. Gait speed is a responsive measure of physical performance for patients undergoing short-term rehabilitation. Gait & Posture 36 (1), 61–64.
DOI: 10.1016/j.gaitpost.2012.01.002

Beckhoff Automation GmbH, 2022. TwinCAT 3 | Kinematic Transformation. TF5110 - TF5113. Verl, Germany.

Beckhoff Automation GmbH, 2025. TwinCAT 3 C/C++. Verl, Germany.

Bieber, J., Pallmer, S., Beitelschmidt, M., Nov. 2023. Computationally efficient implementation of the Gauss–Newton method for solving the forward kinematics of redundant cable-driven parallel robots. PAMM 23 (3).
DOI: 10.1002/pamm.202300231

Bruni, M. F., Melegari, C., De Cola, M. C., Bramanti, A., Bramanti, P., Calabrò, R. S., Feb. 2018. What does best evidence tell us about robotic gait rehabilitation in stroke patients: A systematic review and meta-analysis. Journal of Clinical Neuroscience 48, 11–17.
DOI: 10.1016/j.jocn.2017.10.048

Garrido, J., Riveiro Fernández, E., Silva Muñiz, D., Do Olmo Otero, D., Jul. 2024. Rehabilitación médica mediante Robótica (CDPR) y Realidad Virtual: Proyecto VirtualR3. Jornadas de Automática (45).
DOI: 10.17979/ja-cea.2024.45.10812

Garrido Campos, J., Silva Muñiz, D., Riveiro Fernández, E., Rivera Andrade, J. R., Jul. 2025. Integrated simulation and control environment for the development and safe start-up of cable driven parallel robots. International Journal of Computer Integrated Manufacturing 38 (7), 876–892.
DOI: 10.1080/0951192X.2024.2348499

Gonçalves, R. S., Alves, T., Carbone, G., Ceccarelli, M., 2020. Cable-Driven Robots in Physical Rehabilitation: From Theory to Practice. In: Habib, M. K. (Ed.), Advances in Computational Intelligence and Robotics. IGI Global, pp. 52–96.
DOI: 10.4018/978-1-7998-1382-8.ch003

Hamzeheinejad, N., Roth, D., Monty, S., Breuer, J., Rodenberg, A., Latoschik, M. E., Mar. 2021. The Impact of Implicit and Explicit Feedback on Performance and Experience during VR-Supported Motor Rehabilitation. In: 2021 IEEE Virtual Reality and 3D User Interfaces (VR). IEEE, Lisboa, Portugal, pp. 382–391.
DOI: 10.1109/VR50410.2021.00061

Li, Y., Huang, J., Tian, F., Wang, H.-A., Dai, G.-Z., Feb. 2019. Gesture interaction in virtual reality. Virtual Reality & Intelligent Hardware 1 (1), 84–112.
DOI: 10.3724/SP.J.2096-5796.2018.0006

Luque-Moreno, C., Ferragut-Garcías, A., Rodríguez-Blanco, C., Heredia-Rizo, A. M., Oliva-Pascual-Vaca, J., Kiper, P., Oliva-Pascual-Vaca, Á., 2015. A Decade of Progress Using Virtual Reality for Poststroke Lower Extremity Rehabilitation: Systematic Review of the Intervention Methods. BioMed Research International 2015, 1–7.
DOI: 10.1155/2015/342529

Mikolajczyk, T., Ciobanu, I., Badea, D. I., Iliescu, A., Pizzamiglio, S., Schauer, T., Seel, T., Seiciu, P. L., Turner, D. L., Berteanu, M., Jul. 2018. Advanced technology for gait rehabilitation: An overview. Advances in Mechanical Engineering 10 (7), 1687814018783627.
DOI: 10.1177/1687814018783627

Oyman, E. L., Korkut, M. Y., Ylmaz, C., Bayraktaroglu, Z. Y., Arslan, M. S., Apr. 2021. Design and control of a cable-driven rehabilitation robot for upper and lower limbs. Robotica, 1–37.
DOI: 10.1017/S0263574721000357

PLCopen, 2008. Function Blocks for motion control: Part 4 –Coordinated Motion. Technical Paper, Zaltbommel, The Netherlands.

Shoaib, M., Asadi, E., Cheong, J., Bab-Hadiashar, A., 2021. Cable Driven Rehabilitation Robots: Comparison of Applications and Control Strategies. IEEE Access 9, 110396–110420.
DOI: 10.1109/ACCESS.2021.3102107

Silva, D., Garrido, J., Riveiro, E., Aug. 2022. Stewart Platform Motion Control Automation with Industrial Resources to Perform Cycloidal and Oceanic Wave Trajectories. Machines 10 (8), 711.
DOI: 10.3390/machines10080711

Van der Loos, H. F. M., Reinkensmeyer, D. J., 2008. Rehabilitation and Health Care Robotics. In: Siciliano, B., Khatib, O. (Eds.), Springer Handbook of Robotics. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 1223–1251.

Zuccon, G., Doria, A., Rosati, G., Johnson, C. A., McEligot, L., Hertz, K., Fernan, K., Khan, I., Reggie Edgerton, V., Reinkensmeyer, D. J., Nov. 2024. Design of a Cable-Suspended Robot for Early Stage Gait Rehabilitation. IEEE Transactions on Medical Robotics and Bionics 6 (4), 1616–1626.
DOI: 10.1109/TMRB.2024.3468381