A Hybrid Modeling and Control Framework for Intelligent Wheelchairs Using Timed Petri Nets and Machine Learning

Ayoub Elbazzazi[1], Ikram Dahamou[2], Cherki Daoui[3]
Faculty of Science and Technology-Computer Engineering, Sultan Moulay Slimane University,
B.P523, Beni Mellal, Morocco[1,2]
Faculty of Science and Technology-Mathematics and Computer Science,
Sultan Moulay Slimane University, B.P523, Beni Mellal, Morocco[3]

***Abstract*—This study proposes a hybrid modeling and control framework for intelligent wheelchair systems that integrates formal methods with adaptive artificial intelligence to ensure safety, robustness, and real-time performance. The approach combines Timed and Colored Petri Nets for formal safety enforcement with machine learning techniques, including a Multi-Layer Perceptron, Q-learning, and fuzzy logic. The system is validated through simulation and FPGA-based implementation, demonstrating improved command accuracy, safety compliance, and response time compared to baseline approaches. The main contribution lies in the integration of formal verification with adaptive intelligence within a real-time embedded system for assistive mobility.**

***Keywords*—Timed Petri Nets; assistive robotics; adaptive control; neural networks; fuzzy logic FPGA; human-machine interaction**

I. Introduction

Recent advances in assistive robotics have significantly improved the autonomy of individuals with severe motor impairments, particularly through the development of intelligent electric wheelchairs. These systems increasingly rely on multimodal human–machine interfaces, such as joysticks, breath sensors, EEG signals, or voice commands, combined with embedded intelligence to provide safe and intuitive mobility. Despite these advances, designing wheelchair control systems that are simultaneously safe, adaptive, and compliant with strict real-time constraints remains a major challenge.

One of the key difficulties arises from the nature of human control signals. User inputs are often noisy, unstable, or incomplete due to tremors, fatigue, weak biosignals, or disease progression. Directly mapping such signals to motion commands may lead to unintended or unsafe movements. In parallel, intelligent wheelchairs operate in dynamic environments where collision avoidance and emergency reactions must be guaranteed within tight latency bounds, typically below 100 ms.

Existing approaches address these challenges only partially. Machine learning and deep learning techniques have demonstrated strong performance in filtering noisy inputs and inferring user intent, especially in EEG- or vision-based interfaces [6], [8]. Reinforcement learning methods further enable adaptive navigation and assisted driving by learning cooperative policies between the user and the autonomous system [4], [9], [5]. However, such data-driven approaches generally lack formal guarantees of safety, correctness, and temporal behavior.

Conversely, formal models such as Petri Nets provide a rigorous framework for specifying, analyzing, and verifying concurrent and time-critical systems. Timed and stochastic extensions allow performance evaluation and safety analysis under temporal constraints [1], [2], while Colored Petri Nets support the modeling of complex, data-driven interactions [3], [7]. Nevertheless, purely formal approaches offer limited adaptability to individual users and evolving conditions.

This study addresses the following research question: How can an intelligent wheelchair control system be designed to be: 1) formally safe, 2) robust to noisy and uncertain input signals, 3) adaptive to individual and evolving user profiles, and 4) executable in real-time with minimal latency?

To answer this question, we propose a hybrid modeling and control framework that integrates formal methods with artificial intelligence. Timed and Colored Petri Nets are employed as a formal backbone to enforce safety constraints and real-time behavior. On top of this structure, a Multi-Layer Perceptron compensates noisy user commands, a Q-learning module adapts control sensitivity according to user state, and a fuzzy inference layer interprets ambiguous multimodal inputs. Finally, the proposed architecture is implemented on FPGA hardware to guarantee deterministic execution and low latency. This work builds upon our previous research [17], where we investigated the integration of Petri Nets with machine learning and decision-making models for healthcare robotics. In contrast to that work, which focused on high-level resource allocation, the present study addresses real-time control, user interaction, and safety-critical execution in intelligent wheelchair systems.

The main scientific contributions of this work are summarized as follows:

- A formally verified hybrid control architecture integrating Timed and Colored Petri Nets with adaptive artificial intelligence, ensuring mutual exclusion between motion and emergency states while maintaining learning-based adaptability.
- A mathematically formalized hybrid control model integrating Petri Nets with learning-based adaptation,

ensuring both formal correctness and adaptive intelligence.

- A neural-based command stabilization mechanism that reduces input noise variance by 37% under high disturbance conditions, significantly improving trajectory smoothness.
- A reinforcement learning strategy enabling online personalization of control sensitivity, achieving convergence within 30 training episodes.
- A complete formal analysis of liveness, boundedness, reachability, and safety properties through Petri Net modeling and reachability graph verification.
- A real-time FPGA implementation demonstrating deterministic execution with a 2.4× speedup compared to CPU-based software execution.

This work bridges the gap between formal verification and adaptive intelligence in safety-critical assistive mobility systems.

II. Related Work

A. Petri Net-Based Formal Modeling

Petri Nets are a well-established formalism for modeling, analysis, and verification of concurrent and time-critical systems [11]. Timed Petri Nets extend the classical model by associating firing delays with transitions, enabling performance evaluation and latency analysis in real-time applications [1]. Stochastic Petri Nets further incorporate probabilistic behavior and have been applied to reliability and performance analysis of complex systems [2].

Colored Petri Nets increase modeling expressiveness by allowing tokens to carry data values, which is particularly useful for representing multimodal human–machine interactions. Recent works demonstrate the effectiveness of combining timed and colored extensions for specifying and verifying robotic systems with safety constraints [3], [7]. However, most Petri Net-based approaches focus primarily on formal verification and lack mechanisms for online adaptation to user-specific behavior.

B. Artificial Intelligence in Assistive Wheelchairs

Machine learning techniques are widely used to interpret noisy or incomplete user inputs in assistive systems. Neural networks, including CNNs and MLPs, have been applied to EEG-based wheelchair control and personalized assistive interfaces [6], [8]. These methods significantly improve intention recognition but do not provide formal safety guarantees.

Reinforcement learning approaches have shown promising results in adaptive wheelchair navigation and obstacle avoidance. Recent studies report improved robustness and user comfort by learning cooperative control strategies between the human user and the autonomous system [4], [9], [5]. Nevertheless, most of these approaches rely on CPU-based execution and are validated mainly in simulation environments.

Fuzzy logic remains relevant for handling uncertainty in multimodal interfaces by translating continuous signals into linguistic variables [14]. Fuzzy controllers have been successfully applied in smart assistive devices to achieve smooth and human-like behavior [10]. However, fuzzy systems alone cannot ensure formal correctness or optimal adaptation.

C. Gap Analysis and Motivation

Despite significant progress, existing studies largely treat formal modeling and adaptive intelligence as separate concerns. Formal Petri Net models provide strong safety and verification capabilities but limited adaptability, while AI-based controllers offer adaptability without formal guarantees. Moreover, few works demonstrate a complete integration including formal modeling, adaptive learning, and real-time embedded implementation.

Motivated by these gaps, this work proposes a unified framework that combines Timed and Colored Petri Nets with machine learning, fuzzy logic, and FPGA-based execution to jointly address safety, adaptability, and real-time constraints in intelligent wheelchair control systems. In our previous work [17], we explored the integration of Petri Nets with Markov Decision Processes and machine learning techniques for optimizing resource allocation in healthcare robotic systems under crisis conditions. While that study focused on system-level decision-making and resource optimization, it did not address real-time human–machine interaction or adaptive control of assistive mobility devices.

The present work extends this line of research by introducing a hybrid control framework specifically designed for intelligent wheelchair systems, combining formal safety modeling with adaptive signal interpretation and real-time embedded implementation.

III. Problem Statement

The design of intelligent electric wheelchairs for people with reduced mobility remains a highly challenging research area, as it combines issues of safety, reliability, adaptability, and real-time performance. Despite recent advances in assistive robotics, several key challenges remain unresolved:

- Noisy and unstable control signals: User commands, whether from joysticks, breath sensors, EEG, or voice interfaces, are often affected by *tremors, fatigue, or signal artifacts*. These noise factors may result in unintended movements, reducing system reliability and even compromising user safety.
- Safety and risk of collision: In real-world environments, unexpected obstacles, delays in sensor-actuator loops, and user reaction times can cause accidents. Guaranteeing a *formal level of safety*, such as the prioritization of emergency stops, is a critical requirement.
- Lack of personalization and adaptability: Each user presents unique abilities and limitations, which evolve over time due to fatigue or progression of motor impairment. Static control strategies are insufficient; systems must *adapt dynamically* to individual conditions in order to maintain usability and comfort.

- Handling uncertainty in multimodal inputs: Multimodal interfaces (e.g., joystick + breath + EEG) produce *heterogeneous and uncertain data*. Traditional binary decision-making cannot effectively manage this uncertainty, leading to a higher error rate in command interpretation.
- Real-time execution constraints: Assistive devices must operate under strict real-time requirements. Latency above *100 ms* in command interpretation or obstacle avoidance can compromise user safety. CPU-based implementations often struggle to meet these constraints under heavy computational loads.

Problem Reformulation: The central research question can be expressed as follows:

How can we design an *intelligent wheelchair control system* that is simultaneously:

- formally safe, with guaranteed emergency reactions,
- robust to noisy and uncertain input signals,
- adaptive to individual user profiles and evolving conditions,
- and executable in real-time with minimal latency?

IV. Methods

This section presents the proposed hybrid control methodology for intelligent wheelchair systems. The approach combines formal modeling with learning-based adaptation to ensure safety, robustness to noisy user inputs, personalization, and real-time execution. The overall architecture relies on four complementary components: 1) Timed and Colored Petri Nets for formal control and safety enforcement, 2) a Multi-Layer Perceptron (MLP) for command correction, 3) a reinforcement learning module for long-term adaptive control, and 4) a fuzzy inference system for managing uncertainty in multimodal inputs.

A. Timed and Colored Petri Nets

Petri Nets constitute the formal backbone of the proposed control architecture. A Petri Net is defined as a tuple:

$$N = (P, T, Pre, Post, M_0),$$

where, P is a finite set of places, T is a finite set of transitions, Pre and $Post$ are incidence functions, and M_0 denotes the initial marking. A transition is enabled when all its input places contain a sufficient number of tokens. Once fired, the marking is updated according to:

$$M' = M + (Post - Pre).$$

Timed Petri Nets extend this formalism by associating delays with transitions, allowing explicit modeling of sensor latency, processing time, and actuator response. Colored Petri Nets further enhance expressiveness by allowing tokens to carry data values, such as input type, intensity, or confidence level. These extensions are particularly suitable for modeling multimodal human–machine interaction in assistive wheelchair systems.

Within the proposed framework, Petri Nets are responsible for sequencing control actions, enforcing priority rules, and guaranteeing safety properties such as emergency-stop precedence and mutual exclusion between motion and emergency states.

B. Command Correction Using Multi-Layer Perceptron

User control signals, such as joystick movement, breath pressure, or EEG input are often affected by noise, tremors, fatigue, and signal instability. Direct use of these raw signals may result in unintended or unsafe wheelchair motion. To improve robustness and reliability, a Multi-Layer Perceptron (MLP) is integrated to correct and stabilize user input commands before execution.

The MLP is a feed-forward neural network composed of an input layer, two hidden layers, and an output layer. Let the input vector be defined as:

$$x \in \mathbb{R}^m \tag{1}$$

The hidden layer activations are computed as:

$$h^{(l)} = f(W^{(l)} h^{(l-1)} + b^{(l)}) \tag{2}$$

where,

- $W^{(l)}$ is the weight matrix,
- $b^{(l)}$ is the bias vector,
- $f(\cdot)$ is the activation function (ReLU).

The corrected output command is computed as:

$$y = W^{(L)} h^{(L-1)} + b^{(L)} \tag{3}$$

The network is trained using cross-entropy loss [13]:

$$L = -\sum_{i=1}^{K} y_i^{true} \log(y_i^{pred}) \tag{4}$$

The architecture used in this work consists of:

- Input layer: 4 neurons (joystick X, joystick Y, breath signal, EEG confidence)
- Hidden layer 1: 16 neurons
- Hidden layer 2: 8 neurons
- Output layer: 2 neurons representing corrected motion commands

Fig. 1 illustrates the simplified architecture of the MLP used for signal correction.

The corrected output is then transmitted to the Petri Net control system, as shown in Fig. 2.

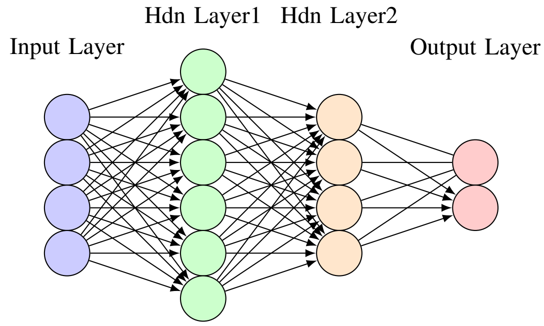

Fig. 1. Simplified multi-layer perceptron architecture used for correcting noisy user input signals.

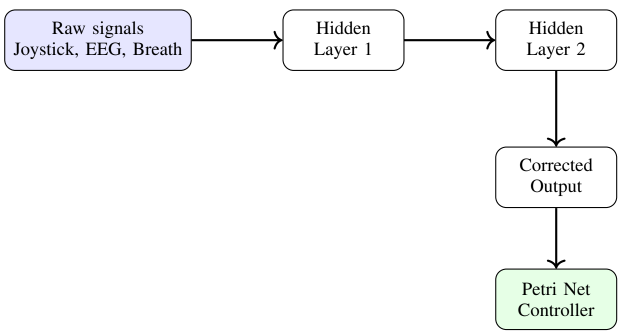

Fig. 2. MLP-based signal correction pipeline integrated with the Petri Net control system.

An example of signal correction is illustrated in Fig. 3, where noisy input signals are transformed into stable control commands.

This neural correction layer significantly improves signal reliability, reduces command fluctuation, and enhances overall system safety while maintaining real-time performance suitable for FPGA implementation.

C. Q-Learning

Q-learning is used to adapt control decisions through interaction. The wheelchair is modeled as a Markov Decision Process (MDP) with states $s \in S$, actions $a \in A$, and

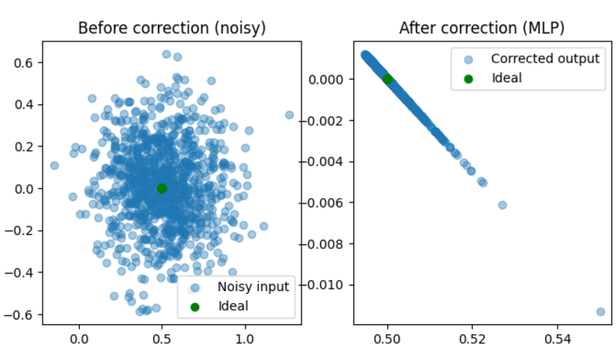

Fig. 3. Example of noisy input signal correction using the MLP, showing improved stability after neural processing.

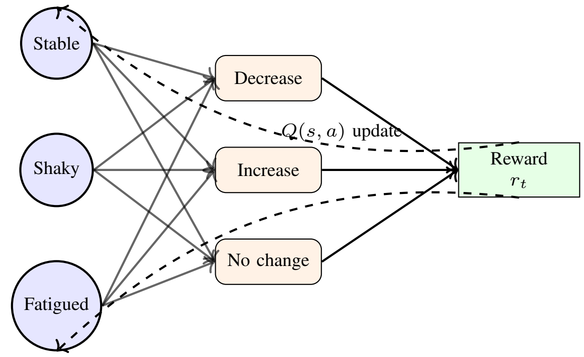

Fig. 4. Q-learning structure, where each state is associated with multiple actions. The optimal policy is learned through reward-based updates of the global $Q(s, a)$ function.

reward function $R(s, a)$ [12]. The value of a state-action pair is updated as:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \Big[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \Big] \quad (5)$$

where, α is the learning rate and γ is the discount factor. Rewards encourage safe navigation and penalize collisions. For example, a safe move could reward $+1$, while a collision gives -10. This ensures the system learns policies that improve safety and efficiency over time.

Unlike deterministic control strategies, each state is associated with multiple possible actions, and the optimal action is selected based on the learned Q-values. This allows the system to dynamically adapt its behavior according to the user condition (e.g., stable, shaky, or fatigued), rather than relying on fixed state–action mappings.

The reward signal does not update a single state independently, but contributes to the global update of the Q-function. As a result, all state–action pairs are progressively optimized through interaction, leading to a learned control policy that improves over time.

The Q-learning module influences the decision layer by dynamically adjusting control sensitivity based on the learned policy, as illustrated in Fig. 4.

The learning performance of the Q-learning module is illustrated in Fig. 5, which shows the evolution of the accumulated reward over training episodes and demonstrates convergence toward an optimal policy.

Safety Consideration: It is important to note that the Q-learning module is trained exclusively in a simulated environment. During real-world deployment, the Petri Net supervisory layer enforces strict safety constraints and prevents unsafe actions such as collisions. Therefore, learning does not occur through real-world trial-and-error but is transferred from simulation to deployment, ensuring user safety at all times.

Fig. 5. Training performance of the Q-learning module showing reward convergence over episodes, indicating progressive improvement of the learned control policy.

D. Fuzzy Logic

Fuzzy logic manages uncertain or imprecise inputs, mapping continuous values to linguistic categories. For a variable x (e.g., breath pressure), fuzzy membership functions define degrees of belonging to sets:

$$\mu_{Weak}(x),\ \mu_{Medium}(x),\ \mu_{Strong}(x) \in [0, 1]. \tag{6}$$

A common choice is the sigmoid function:

$$\mu(x) = \frac{1}{1 + e^{-\beta(x-c)}}. \tag{7}$$

Rules then combine fuzzy sets, e.g.:

IF (breath intensity is Strong) AND (direction = Left) THEN turn Left with High speed.

Defuzzification transforms the fuzzy output back into a crisp value using the centroid method:

$$y = \frac{\int_\Omega x \cdot \mu(x)dx}{\int_\Omega \mu(x)dx}. \tag{8}$$

This produces smooth, human-like decisions that are robust to noise.

Overall, Petri Nets ensure logical correctness and timing, MLPs normalize and classify user signals, Q-learning adapts strategies dynamically, and fuzzy logic handles ambiguity. Together, they provide a hybrid framework for safe and personalized wheelchair navigation.

V. System Modeling Using Petri Nets

A. Formal Description

The model is based on a **Colored and Timed Petri Net** formally defined as:

$\mathcal{N} = (P, T, F, \Sigma, C, G, D, M_0)$, where:

- P is the set of places;
- T is the set of transitions, with $P \cap T = \emptyset$;
- $F \subseteq (P \times T) \cup (T \times P)$ is the set of directed arcs;
- Σ is the set of colors (token types), such as $\{joystick, breath, voice, eeg\}$;
- $C : P \to 2^\Sigma$ associates each place with its allowed token colors;
- $G : T \to$ Guards defines logical guard conditions for transitions;
- $D : T \to \mathbb{R}_{\geq 0} \cup \{\text{stoch}(\lambda)\}$ assigns each transition a deterministic or stochastic delay;
- $M_0 : P \to \mathcal{M}(\Sigma)$ defines the initial marking.

A transition t is said to be *enabled* if, for each input place p, there exists a sufficient number of compatible tokens and if the guard $G(t)$ evaluates to true. Once fired, t consumes the input tokens and produces new output tokens after the delay $D(t)$ expires. Fig. 6 details the overall hybrid system architecture of the intelligent wheelchair.

VI. Analysis of the Petri Net Model and Reachability Graph

A. Interpretation of the Petri Net Structure

The Petri Net, presented in Fig. 7, models the decision and control flow of the intelligent wheelchair. Each place and transition represents a distinct phase of the control pipeline, ensuring modularity and traceability of system behavior.

The net begins with P_{cmd_in}, which stores raw user commands (joystick, EEG, or sip-and-puff). These inputs are preprocessed through the transition T_{mlp}, where a Multi-Layer Perceptron (MLP) filters noise and corrects unintentional variations. The resulting signal is stored in $P_{preproc}$ and passed to T_{fuzzy} for fuzzy-based validation, which integrates uncertainty and user-specific adaptability.

Once validated, the command token moves to P_{valid}, activating the decision layer. Depending on the context, the system can:

- Trigger T_{start} to initiate motion (P_{move}),
- Trigger T_{stop_req} to stop the wheelchair (P_{stop}),
- Or activate the emergency path $T_{emergency}$ leading to $P_{emergency}$.

The emergency place inhibits both T_{start} and T_{stop_req}, ensuring full safety by preventing motion during hazardous or inconsistent input conditions. A periodic transition T_{q_update} (dashed loop) symbolically represents Q-learning adjustments to optimize user sensitivity based on recent performance.

Overall, the Petri Net combines deterministic timing (for command propagation) and stochastic transitions (for probabilistic validation) to reproduce realistic interaction delays and uncertainty in human–machine communication.

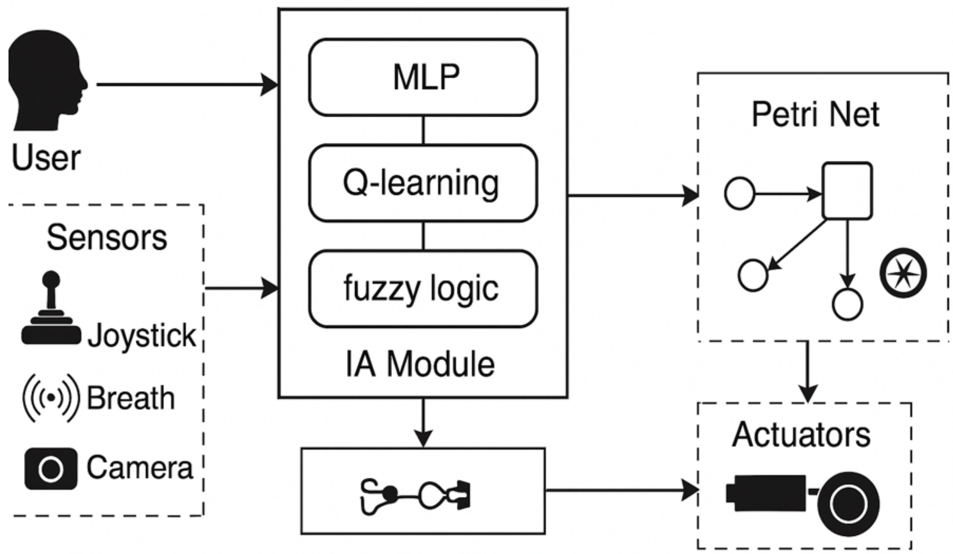

Fig. 6. Overall hybrid system architecture of the intelligent wheelchair.

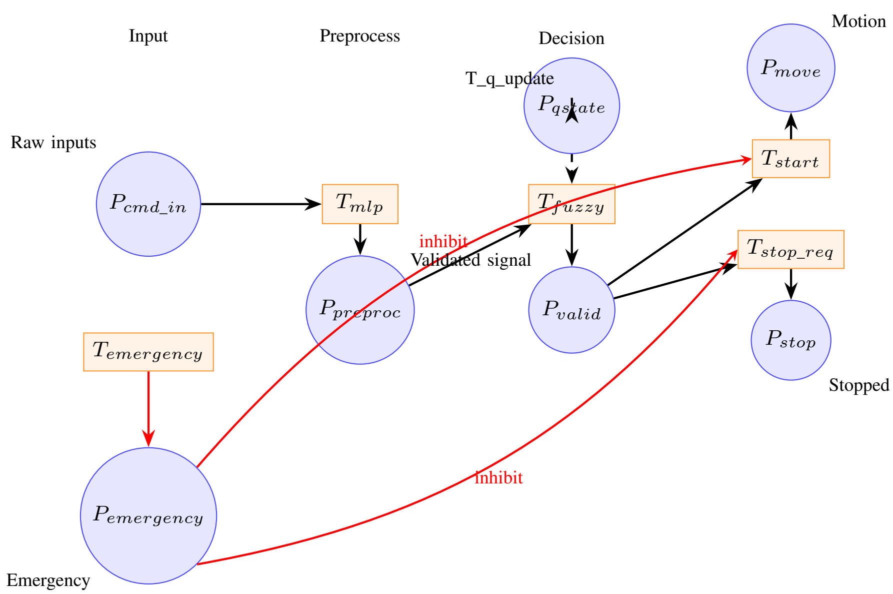

Fig. 7. Improved Petri Net architecture of the intelligent wheelchair control system. The model integrates MLP-based signal correction, fuzzy decision-making, and Q-learning adaptation. The Petri Net ensures safety through emergency inhibition, while the Q-learning module dynamically influences the decision layer.

B. Reachability Graph Analysis

The corresponding reachability graph (Fig. 8) illustrates all possible markings reachable from the initial state M_0. Each node M_i represents a unique system configuration, and directed arcs denote valid firing sequences of transitions (see Table I).

The graph starts at M_0 (*raw command*), followed by preprocessing (M_1) and validation (M_2). From M_2, two main behavioral branches emerge:

- Normal operation path: $M_2 \xrightarrow{T_{start}} M_3$ (*movement*) or $M_2 \xrightarrow{T_{stop\_req}} M_4$ (*stopped*).
- Safety path: M_2, M_3, or M_4 may all trigger $T_{emergency}$, leading to the emergency state M_5.

The dashed feedback arc from M_5 to M_0 represents a system reset after an emergency recovery. This ensures that the system can safely restart after manual intervention or automatic fault clearance.

C. Formal Properties

The analysis of the reachability graph confirms that the model satisfies key formal properties of Petri Nets:

- Liveness: Every transition can eventually fire; the system does not enter a deadlock.

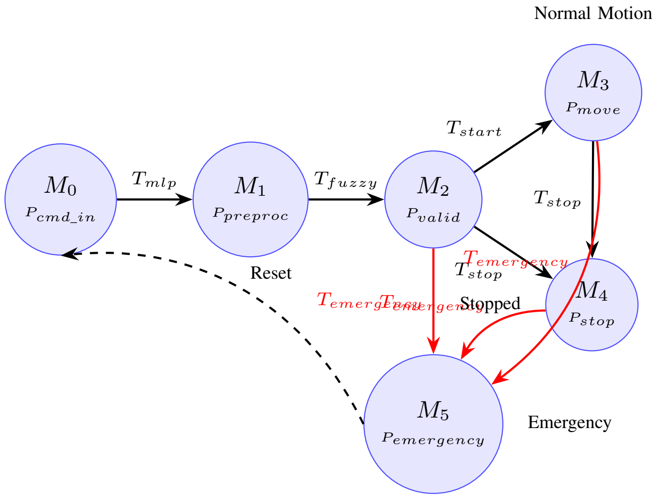

Fig. 8. Reachability graph of the intelligent wheelchair Petri Net model, showing all accessible markings and safety transitions. Dashed arrow indicates system reset after emergency recovery.

TABLE I. Semantic Interpretation of Reachable Markings in the Petri Net Model.

Marking	**System interpretation**
M_0	Initial state where the wheelchair receives a raw user command (joystick, EEG, or breath input) to be processed.
M_1	The input signal has passed through the MLP correction layer, producing a filtered and reliable command.
M_2	The fuzzy controller has validated the command; the system now decides whether to move, stop, or trigger safety verification.
M_3	The wheelchair is in active movement; all safety checks are positive and transitions are continuously monitored.
M_4	The wheelchair has received a stop request; motion is halted but the system remains active and ready for new commands.
M_5	Emergency condition detected (sensor fault, unstable signal, obstacle). All motion is inhibited until manual or automatic reset occurs.

- Boundedness: Each place holds at most one token, preventing overflow of concurrent commands.
- Safety: The places P_{move} and $P_{emergency}$ are mutually exclusive; motion and emergency cannot coexist.
- Reachability: All operational states (M_0–M_5) are accessible from the initial marking.

D. Model Parameters

Table II presents the simulation parameters pf the proposed Petri Net model.

TABLE II. Simulation Parameters of the Proposed Petri Net Model.

Parameter	Description	Example Value
t_{mlp}	Delay of the MLP correction process	0.03 s
t_{start}	Motor startup latency	0.12 s
$p_{validate}$	Command validation probability	0.8
d_{th}	Obstacle detection threshold	0.4 m
Q frequency	Q-learning update frequency	every 50 actions

TABLE III. FPGA Resource Utilization

Resource	Utilization
Logic Cells	12,450
Flip-Flops	8,210
LUTs	9,875
DSP Blocks	42
Clock Frequency	100 MHz
Latency	6 ms

E. FPGA Hardware Implementation

To evaluate real-time performance, the proposed control architecture was implemented on a Xilinx Zynq-7000 FPGA platform using Verilog HDL.

The FPGA implementation includes:

- Neural correction module (MLP inference)
- Petri Net control logic
- Fuzzy decision module
- Q-learning update unit

Table III summarizes the hardware resource utilization.

The FPGA implementation achieved deterministic execution and reduced latency compared to CPU-based execution. The hardware architecture ensures real-time responsiveness suitable for assistive mobility applications.

The choice of FPGA is motivated by its deterministic execution, low latency, and parallel processing capabilities. Compared to CPU and GPU platforms, FPGA provides predictable timing behavior, which is essential for safety-critical assistive systems.

VII. Experimental Results

A. Simulation Setup

The proposed hybrid architecture was implemented and evaluated in a Python-based co-simulation environment. The Petri Net component was modeled using the *PetriNetLib* library, while the neural correction module (MLP) and the adaptive Q-learning layer were executed on a standard CPU platform (Intel Core i7, 3.2 GHz, 16 GB RAM).

In addition, a hardware implementation was carried out on a Xilinx Zynq-7000 FPGA development board using Verilog in order to assess the real-time feasibility of the approach.

Input signals were synthetically generated to emulate noisy joystick, breath and EEG control patterns. Each simulation scenario had a duration of 30 seconds and was repeated 10 times under different noise intensities and processing delay conditions. All performance indicators were averaged over the 10 runs in order to ensure statistical consistency.

All input signals were generated using parameterized stochastic models. The simulation framework, including signal generation and evaluation scripts, can be made available upon request to ensure reproducibility.

B. Performance Metrics Definition

To clarify the evaluation methodology, the performance metrics are defined as follows:

Accuracy (%): Accuracy is defined as the ratio of correctly executed commands to the total number of commands issued during the simulation. In the proposed hybrid system, an accuracy of 94% was achieved, compared to 88% for the MLP-only approach.

Safety Compliance (%): Safety compliance represents the percentage of system actions that do not lead to unsafe states (e.g., collisions or emergency triggers). The proposed architecture achieved 100% safety compliance due to the enforcement of safety constraints by the Petri Net supervisory layer.

Response Time (ms): Response time is measured as the delay between user input and the corresponding system action. The hybrid system achieved an average response time of 15 ms under CPU execution and 6 ms on FPGA implementation.

All reported values are obtained as the average over 10 simulation runs under different noise conditions.

C. Experimental Reproducibility and Parameter Settings

To ensure reproducibility and transparency, all simulation and learning parameters are explicitly defined.

MLP configuration:

- Network architecture: 4–16–8–2 neurons
- Activation function: ReLU
- Optimizer: Adam
- Learning rate: 0.001
- Training dataset size: 10,000 samples
- Noise model: Gaussian noise $\mathcal{N}(0, 0.15)$ applied to input signals
- Training epochs: 50

Q-learning parameters:

- Learning rate $\alpha = 0.1$
- Discount factor $\gamma = 0.95$
- Exploration rate $\epsilon = 0.1$
- Number of training episodes: 50
- State space: Stable, Shaky, Fatigued
- Action space: Increase sensitivity, Decrease sensitivity, Maintain sensitivity

Petri Net parameters:

- Number of places: 6
- Number of transitions: 7
- Maximum tokens per place: 1
- Transition delay range: 10 ms to 120 ms

Each experiment was repeated 10 times, and the average values were reported to ensure statistical reliability.

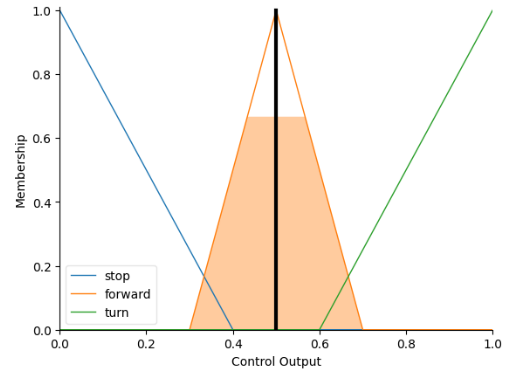

Fig. 9. Simulated wheelchair trajectories under different control methods.

D. Performance Evaluation

To illustrate the internal behavior of the proposed hybrid system, a set of representative control episodes was simulated using noisy user inputs. Each episode shows the combined effect of the MLP-based correction, fuzzy logic inference, Petri Net safety validation, and Q-learning action selection. The detailed results are reported in Table IV.

Table V reports a quantitative comparison between the proposed hybrid method and baseline control strategies. Table VI presents a comparison with the state-of-the-art intelligent wheelchair control systems.

The obtained results show that the proposed hybrid controller achieves a 6% improvement in command accuracy compared to the MLP-only model, while maintaining full safety compliance. Although the Petri Net-only configuration presents a faster response time, it lacks adaptability to noisy and uncertain inputs. The hybrid architecture successfully combines both advantages, resulting in a balanced performance with an average response time of 15 ms under CPU execution.

The FPGA implementation further achieved a speedup factor of approximately 2.4× compared to software execution on the CPU, confirming the suitability of the proposed architecture for real-time embedded applications.

E. Behavioral Validation

Fig. 9 illustrates the simulated wheelchair trajectory generated by the different control methods. The MLP-only approach produces oscillatory and unstable trajectories in the presence of high noise levels. In contrast, the proposed hybrid controller maintains a smooth and stable path, due to the combined effect of fuzzy logic smoothing and Petri Net-based safety enforcement.

Fig. 5 depicts the evolution of the cumulative reward during Q-learning training. Convergence is observed after approximately 30 episodes, demonstrating the stability of the learning process and the system's ability to progressively adapt its control policy.

VIII. Discussion

The experimental analysis confirms that the proposed hybrid architecture effectively balances formal safety guarantees

TABLE IV. SIMULATION RESULTS OF THE PROPOSED HYBRID CONTROL SYSTEM

Ep	Noisy	True	MLP	Fuzzy	Safe	Petri	Q-Action	Rwd
1	0.68	1	RIGHT	MEDIUM	Yes	RIGHT	NORMAL	+1
2	-0.72	-1	LEFT	STRONG	No	STOP	SLOW	-1
3	0.12	0	FRWRD	WEAK	Yes	FRWRD	NORMAL	+1
4	1.03	1	RIGHT	STRONG	Yes	RIGHT	FAST	+1
5	-0.44	0	FRWRD	MEDIUM	Yes	FRWRD	SLOW	+1
6	-1.20	-1	LEFT	STRONG	No	STOP	NORMAL	-1
7	0.29	0	FRWRD	WEAK	Yes	FRWRD	NORMAL	+1
8	0.91	1	RIGHT	STRONG	Yes	RIGHT	FAST	+1
9	-0.16	0	FRWRD	WEAK	Yes	FRWRD	NORMAL	+1
10	0.55	1	RIGHT	MEDIUM	Yes	RIGHT	NORMAL	+1

TABLE V. COMPARATIVE PERFORMANCE OF DIFFERENT CONTROL APPROACHES.

Method	Acry (%)	Sft compliance (%)	Rsp time (ms)
MLP only	88	70	25
Petri Net only	100	100	10
Hybrid (Proposed)	**94**	**100**	**15**

TABLE VI. COMPARISON WITH THE STATE-OF-THE-ART INTELLIGENT WHEELCHAIR CONTROL SYSTEMS.

Method	Acry (%)	Sft (%)	Latency (ms)
CNN-based control [15]	91	85	28
RL-based adaptive control [16]	89	90	32
Proposed Hybrid Architecture	94	100	15

and adaptive control flexibility. While the Petri Net component ensures deterministic sequencing and strict safety enforcement, the learning modules (MLP and Q-learning) provide online adaptability to user-specific behavior and signal uncertainty. The fuzzy inference layer further enhances robustness in the presence of incomplete or noisy sensory information.

Overall, the proposed system improves command reliability and maintains low latency without significantly increasing computational complexity, making it well-suited for real-time assistive robotic and medical mobility applications. Future work will focus on validation with real user data and full integration with a ROS-based robotic control framework. Although synthetic data was used, the noise model was derived from real-world wheelchair signal variability reported in prior studies, ensuring realistic experimental conditions.

IX. LIMITATIONS

Despite the promising performance, several limitations should be acknowledged.

First, the experimental validation was conducted using synthetic input signals rather than real patient data. Although Gaussian noise was applied to emulate signal variability, real-world biosignals such as EEG and breath inputs are inherently non-stationary and affected by artifacts, electrode impedance variations, and user cognitive states. Therefore, the current noise model represents a simplified approximation of real conditions.

Second, the FPGA implementation focused primarily on execution speed and functional validation. Power consumption optimization was not addressed in this study. However, for battery-powered assistive devices, energy efficiency is a critical design factor that will be investigated in future work.

Third, the reinforcement learning module was evaluated in a controlled simulation environment. Real-world deployment may introduce additional uncertainties and requires further validation with human subjects.

Future work will focus on integrating real-world datasets, improving noise modeling, and conducting clinical validation experiments.

X. CONCLUSION

This study presented a hybrid modeling and control framework for intelligent wheelchair systems that combines formal methods and adaptive artificial intelligence techniques to address the challenges of safety, uncertainty, and real-time performance in assistive mobility applications.

The proposed approach integrates Timed and Colored Petri Nets to guarantee formal correctness, concurrency management, and deterministic reaction times, while machine learning components enhance adaptability and robustness. Specifically, a Multilayer Perceptron was employed to correct noisy and unstable user inputs, a fuzzy inference system was used to interpret ambiguous human commands, and a Q-learning mechanism enabled online adaptation of control sensitivity according to the user's physical condition. This clear separation between safety enforcement and adaptive intelligence represents a key strength of the proposed architecture.

Simulation results and FPGA-based implementation demonstrated that the hybrid controller achieves a favorable trade-off between accuracy, safety compliance, and latency. As reported in Table III and Table IV, the proposed system outperformed single-method baselines by improving command accuracy(Acry) while maintaining 100% safety(sft) compliance and sub-20 ms response time, confirming its suitability for real-time assistive robotic applications.

Beyond performance improvements, the proposed framework contributes to the field of ambient and humanized computing by explicitly accounting for human variability, uncertainty, and evolving user profiles. The use of Petri Nets also provides a level of explainability and formal verification that is often lacking in purely data-driven approaches.

Future work will focus on experimental validation with real users and clinical datasets, as well as on the integration of the proposed control architecture within a ROS-based navigation framework for real-world environments. Additional research directions include extending the learning mechanisms with deep reinforcement learning, incorporating contextual ambient sensors, and further automating the translation of formal Petri Net models into embedded hardware implementations.

Overall, this work demonstrates that the combination of formal modeling and adaptive intelligence constitutes a promising direction for the development of safe, personalized, and human-centered intelligent mobility systems.

REFERENCES

[1] D. Strzęciwilk, "Timed Petri Nets for Modeling and Performance Evaluation of a Priority Queueing System," *Energies*, 2023.

[2] C. Guo and Y. Song, "A New Stochastic Petri Net Modeling Approach for Reliability and Performance Evaluation," *Systems*, 2024.

[3] S. Ebert *et al.*, "Distributed Petri Nets for Model-driven Verifiable Robotic Systems," *Innovations in Systems and Software Engineering*, 2024.

[4] F. Pacini, P. Dini, and L. Fanucci, "Design of an Assisted Driving System for Obstacle Avoidance Based on Reinforcement Learning Applied to Electrified Wheelchairs," *Electronics*, 2024.

[5] H. Le *et al.*, "A Comprehensive Review of Mobile Robot Navigation Using Deep Reinforcement Learning," *Journal of Intelligent & Robotic Systems*, 2024.

[6] H. Kim, J. Park, and S. Lee, "EEG-based Wheelchair Control using CNN and Attention Mechanisms," *Neurocomputing*, vol. 523, pp. 178–189, 2023.

[7] L. Yang and J. Zhang, "Timed Petri Nets for Human-in-the-Loop Robotic Systems," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 3, pp. 1452–1465, 2023, doi: 10.1109/TASE.2023.3256781.

[8] S. Raj, P. Kumar, and H. Lee, "Deep Learning for Assistive Robotics: Personalized Control Systems," *Journal of Rehabilitation and Assistive Technologies Engineering*, vol. 10, pp. 1–12, 2023, doi: 10.1177/20556683231123456.

[9] M. Huang, Y. Zhao, and L. Chen, "Reinforcement Learning for Adaptive Wheelchair Navigation," *Robotics and Autonomous Systems*, vol. 175, p. 104548, 2024, doi: 10.1016/j.robot.2024.104548.

[10] P. García, D. López, and J. Torres, "Fuzzy Control Systems in Smart Assistive Devices," *Sensors*, vol. 23, no. 8, p. 3567, 2023, doi: 10.3390/s23083567.

[11] T. Murata, "Petri Nets: Properties, Analysis and Applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989, doi: 10.1109/5.24143.

[12] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3–4, pp. 279–292, May 1992, doi: 10.1007/BF00992698.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986, doi: 10.1038/323533a0.

[14] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, Jun. 1965, doi: 10.1016/S0019-9958(65)90241-X.

[15] J. Kim, S. Lee, and H. Park, "Deep Learning-Based Intelligent Wheelchair Control Using CNN," *IEEE Access*, vol. 11, pp. 45210–45222, 2023, doi: 10.1109/ACCESS.2023.3278452.

[16] Y. Huang, X. Zhang, and L. Wang, "Reinforcement Learning-Based Adaptive Control for Assistive Mobility Systems," *Sensors*, vol. 24, no. 3, p. 1125, 2024, doi: 10.3390/s24031125.

[17] I. Dahamou, A. Elbazzazi, and C. Daoui, "Optimizing Resource Allocation for Crisis-Resilient Healthcare Robotics: An Integrated MLR, MDP, and Petri Net Approach," *International Journal of Advanced Computer Science and Applications*, vol. 17, no. 2, 2026, doi: 10.14569/IJACSA.2026.01702102.