SSRG International Journal of Electrical and Electronics Engineering
ISSN: 2348-8379/ https://doi.org/10.14445/23488379/IJEEE-V11I8P116

Volume 11 Issue 8, 177-187, August 2024
© 2024 Seventh Sense Research Group®

Original Article

A Statistical Signal Processing and Machine Learning Approach of Dithered Pseudo Random Noise for Evaluating the Performance of Ring Laser Gyro (RLG)

Thoudoju Sreeramulu[1], Lavadya Nirmala Devi[2], Avunuri Ramchander Rao[3]

[1,2]*Department of Electronics and Communication Engineering, Osmania University, Telangana, India.*
[3]*Directorate of Laser Systems, Research Center Imarat, DRDO, Telangana, India.*

[1]*Corresponding Author : t_sreeram@hotmail.com*

Received: 07 June 2024 Revised: 10 July 2024 Accepted: 08 August 2024 Published: 31 August 2024

Abstract *- This paper presents a new approach to enhance Ring Laser Gyroscopes (RLGs) performance using Statistical Signal Processing (SSP) and Reinforcement Learning (RL). RLGs are devices that are highly significant for most navigation systems; they are, however, inherently subject to the lock-in effect, static and dynamic, which degrades the accuracy and performance. To minimize these issues, the concept of Dynamic Dither Angle (DDA) and Dither Symmetry Angle (DSA) calculations, which are derived from dithered Pseudo Random Noise (PRN), is injected into the RLG system. The DDA is calculated recursively, providing a method to track the cumulative effect of dither pulses. In contrast, the DSA method takes the form of a balance measure of the errors in the system caused by dither. Deep Deterministic Policy Gradient (DDPG) is used as an optimization method for PRN values to avoid lock-in effects for better measurement accuracy. The RL agent works in interaction with the RLG system and tunes the PRN values such that multiple performance metrics can be fed back, comprising lock-in occurrences and angular measurement accuracies. Experimental results proved that both static and dynamic lock-in effects can be reduced enormously, whereby overall gyroscope performance is enhanced. This SSP-RL integration provides a robust and efficient way to advance RLG technology into more reliable solutions for accurate navigation.*

Keywords *- Dither, Navigation, Pseudo Random Noise, Reinforcement Learning, Ring Laser Gyro, Statistical Signal Processing.*

1. Introduction

Ring Laser Gyroscopes (RLGs) are vital devices for modern navigation systems, which operate at elevated precision and stability in angular velocity measurement. However, an essential deficiency of RLGs is the lock-in effect [1], which, in principle, occurs due to a coincidence of two counter-propagating beams at a standard frequency, leading the gyroscope to lose its sensitivity to small angular movements. This effect, encompassing both static and dynamic lock-in [1], drastically decreases the accuracy of RLGs, particularly at low angular rates. Mechanical dithering is often implemented to reduce the lock-in effect [2].

However, standard dithering schemes are not very successful in an absolute sense in eliminating lock-in, particularly dynamic lock-in, which occurs even with dithering. In addition, introducing Pseudo Random Noise (PRN) to the dither drive signal has demonstrated some encouraging results in lowering dynamic lock-in; however, optimising PRN [3] values remains a tough task. Existing techniques fail to fully eliminate dynamic lock-in and often rely on heuristic or trial-and-error approaches for PRN optimization. Moreover, the current methods do not leverage the potential of adaptive and data-driven techniques like machine learning, which can offer more precise and real-time optimization.

This paper further proposes an entirely new approach of combining SSP with RL [5] to improve RLGs [4]. Two new concepts are introduced: Dynamic Dither Angle (DDA) and Dither Symmetry Angle (DSA). Where the DDA is a measure that captures the cumulative effect of the dither pulses in a recursive calculation, the DSA will calculate in a balanced way and thus give the possibility to find and correct lock-in effects. DDPG reinforcement learning techniques are applied to the optimization of PRN values.

The RL agent interacts with the system by making real-time adjustments in PRN values based on the feedback provided by the performance metrics. These performance metrics mainly rely on the occurrence of lock-in and the precision of angular measurements, where dynamic adjustment is usually aimed at minimizing lock-in effects and thereby improving the overall accuracy of the gyroscope.

Contributions of the works are threefold: first, the DDA and DSA methods for lock-in reduction are introduced to enhance the performance of RLGs; secondly, a Reinforcement Learning framework is developed and implemented for PRN value optimization [7]; and finally, the proposed methods are evaluated using simulations, showing significant lock-in effect reductions and gains in gyroscope performance.

The integration of these two tools-SSP and RL-will offer a new, robust solution for advancing RLG technology to make it more reliable and accurate for applications in precision navigation.

2. Related Work

2.1. Existing Methods

Lock-in issues in Ring Laser Gyroscopes (RLGs) have been the topic of much research in recent years. To reduce the lock-in effects, both in static and dynamic states, several methods have been proposed and adopted. This section will review the most notable existing methods and their drawbacks.

The dithering of various mechanical forms was one of the earliest and most widely applied techniques to reduce the lock-in effect in RLGs. The dither, which periodically oscillates the gyroscope, makes the system spend much time far from the zero angular rate region, thus making it less likely that the counter-propagating laser beams will lock. While effective at minimizing the likelihood of lock-in due to static imbalance, mechanical dithering can be less effective against dynamic imbalance, where in some phases of the dither cycle, the beams may still synchronize [8].

Injecting PRN into the dither drive signal has recently been proposed as one approach for breaking this periodicity that leads to dynamic lock-in. PRN injection results in randomizing the dither motion and, as a by-product, de-correlating the frequencies of the laser beams, so it generally diminishes the tendency for lock-in to occur. This can, however, make the actual optimization of PRN parameters highly complex in practice and often quite challenging to achieve optimally, even through time-consuming trial-and-error or heuristic methods.

Different advanced control algorithms, including adaptive control and fuzzy logic [9], have been used for the performance improvement of RLGs. The mentioned techniques adjust the control parameters by the system's behavior, so it is a more reactive technique aimed at diminishing the effect of lock-in. Still, generally, these algorithms have a high order in nature, for example [10], with the consequence that their implementation is highly complex. Their efficiency depends on the underlying model quality, as well as the parameters themselves, which are tuned and could be pretty cumbersome in actual cases.

2.2. Machine Learning in RLGs

This field of research is relatively young but has very rapid growth. The RLG system, with the application of machine learning techniques, presents optimal possibilities for complex nonlinear systems more effectively than conventional methods.

RLG measurement errors have been predicted and corrected using supervised learning [11] approaches. Labelled data can be large-quantity datasets, which enable training models to recognize patterns of lock-in effects and anomalies for pre-emptive adjustments. However, the key issues remain reliance on a lot of labelled data and generalization to unseen conditions, limiting applicability in dynamic environments.

Reinforcement Learning (RL) [12] is exciting for the optimization of PRN values in RLG systems due to how RL learns an optimal policy interacting with the environment. The RL agent tries out different settings for PRN and learns from the feedback it receives about how different task performance methods affect lock-in effects and performance. The latest developments in the class of RL algorithms belong to that of Deep Deterministic Policy Gradient (DDPG) [12].

Despite intensive progress, the existing methods yet to be adopted have limitations in considering dynamic lock-in comprehensively and optimizing the PRN parameters effectively. Mechanical and electromagnetic dithering serve as partial solutions only, with PRN injection showing some promise. However, its optimization problem is complex. Improvements can be made through advanced control algorithms and digital signal processing, but these methods may grow complex and computationally intensive.

In such cases, Machine Learning, and particularly Reinforcement Learning, appears to be a powerful technique to overcome the limitations in applying adaptive, data-driven methods for the optimal setting of PRN values, which will minimize the lock-in effects better than previous ways. This gap will be bridged by integrating statistical signal processing with the DDPG algorithm into one coherent work to come up with a viable solution for enhancing RLG performance.

3. Methodology

3.1. RLG System Description

There are several main components to the RLG system. The dither motor provides a periodic oscillatory motion to the gyroscope to depreciate the lock-in of its axis. A closed optical path is created with counter-propagating beams of a laser. Photo-detectors monitor the interference pattern that these beams produce, from which it is possible to measure angular velocity. To alleviate the lock-in impact, mechanical dithering is applied to the RLG. The dither motor enforces oscillatory motion to take the system off the region of zero angular rate, thereby breaking the synchronization of the laser beams.

3.2. Pseudo-Random Noise (PRN) Injection

To overcome the limitations of the traditional dithering methods, PRN is injected into the dither drive signal. The PRN makes the dither motion random and thus removes the periodicity of the dither. It can be observed that the values of the PRN in Equations 1 and 2 are generated with a pseudo-random number generator and updated dynamically through reinforcement learning to optimize the dither signal.

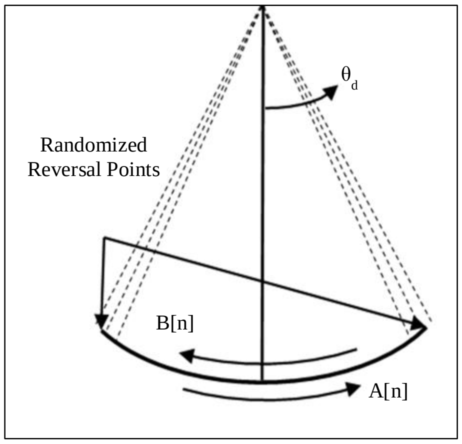

Fig. 1 Pendulum-like system with randomized reversal points

$$\boldsymbol{PRN_A[n] = PRN_A[n-1]} + \Delta \boldsymbol{A[n]} \tag{1}$$

$$\boldsymbol{PRN_B[n] = PRN_B[n-1]} + \Delta \boldsymbol{B[n]} \tag{2}$$

Here, the clockwise and anticlockwise angles [13] caused by the dither motion are given as *A[n]* and *B[n]*, respectively. Figure 1 shows a pendulum-like system similar to RLG with random reversal points showing the *A[n]* and *B[n]* in terms of the dither angle θ_d acquired from the RLG. These *A[n]* and *B[n]* pulses are acquired over a specific duration for further analysis.

3.3. Statistical Signal Processing (SSP) Approach

3.3.1. Dynamic Dither Angle (DDA) Calculation

The DDA is defined by its previous values and alternating additions and subtractions of A[n] and B[n]. If the sequence values are denoted by S[n], then the DDA can be expressed as Equation 3.

$$\{\mathrm{S}[0], \mathrm{S}[1], \mathrm{S}[2], \ldots\} = \left\{\frac{-\mathrm{A}[0]}{2}, \frac{-\mathrm{A}[0]}{2} + \mathrm{A}[0], \ \frac{-\mathrm{A}[0]}{2} + \mathrm{A}[0] - \mathrm{B}[0], \frac{-\mathrm{A}[0]}{2} + \mathrm{A}[0] - \mathrm{B}[0] - \mathrm{A}[1], \ldots\right\} \tag{3}$$

It can noticed that these terms are being added and subtracted alternately, starting from the addition of A[n] followed by a subtraction of B[n]. The general form for S[n] can be written depending on whether n is even or odd.

$$\mathrm{S}[2\mathrm{k}] = \frac{-\mathrm{A}[0]}{2} + \sum_{\mathrm{i}=0}^{\mathrm{k}-1}(\mathrm{A}[\mathrm{i}] - \mathrm{B}[\mathrm{i}]) \quad \text{for even } \mathrm{n} = 2\mathrm{k} \tag{4}$$

The sequence comprises an even number of additions and subtractions.

$$\mathbf{S[2k+1]} = \frac{\mathbf{-A[0]}}{\mathbf{2}} + \sum_{\mathbf{i=0}}^{\mathbf{k}}(\mathbf{A[i]}) - \sum_{\mathbf{i=0}}^{\mathbf{k-1}}(\mathbf{B[i]}) \quad \textbf{for odd n = 2k + 1} \tag{5}$$

Similarly, the above sequence comprises an odd number of additions and subtractions. Equations 4 and 5 can be combined into a single expression, and it can be written as,

$$\mathbf{S[n]} = \frac{\mathbf{-A[0]}}{\mathbf{2}} + \sum_{\mathbf{i=0}}^{\lfloor \frac{\mathbf{n}}{\mathbf{2}} \rfloor}(\mathbf{A[i]}) - \sum_{\mathbf{i=0}}^{\left\lfloor \frac{\mathbf{n-1}}{\mathbf{2}} \right\rfloor}(\mathbf{B[i]}) \tag{6}$$

Where ⌊x⌋ represents the floor function that represents the highest integer value that is smaller than or equal to x.

3.3.2. Dither Symmetry Angle (DSA) Calculation

The values in the DSA represent a good compromise in representing the dither-induced errors. Let us consider the sequence values x_n, where n is the step in the sequence.

$$\{x_0, x_1, x_2, \ldots\} = \ \{0, \frac{A[0]-B[0]}{2}, \frac{A[0]-B[0]}{2} + \frac{A[1]-B[0]}{2}, \ \frac{A[0]-B[0]}{2} + \frac{A[1]-B[0]}{2} + \frac{A[1]-B[1]}{2}, \ldots\} \tag{7}$$

This will go by adding $(\mathrm{A[i]} - \mathrm{B[j]})/2$ where i is the current index of A and j is the index of *B* from two steps back. It can be seen that every time the index is even, the current *A* and *B* of $\lfloor(\mathrm{n}-2)/2\rfloor$ are used, and every time the index is odd, the same A is used as the last step and the current *B*.

To write in general term x_n, it should be made clear how *A* and *B* are indexed.

$$x_{2k} = x_{2k-1} + \frac{A[k]-B[k-1]}{2} \ for \ even \ n = 2k \tag{8}$$

$$x_{2k+1} = x_{2k} + \frac{A[k]-B[k]}{2} \ for \ odd \ n = 2k+1 \tag{9}$$

Thus, the general form for x_n, from Equations 8 and 9, can be written as follows:

$$\begin{aligned} x_n &= 0 && for \ n = 0 \\ &= \ x_{n-1} + \frac{A_{\lceil \frac{n}{2} \rceil} - B_{\lfloor \frac{n-1}{2} \rfloor}}{2} && for \ n > 0 \end{aligned} \tag{10}$$

Where $\lceil n/2 \rceil$ is ceiling function, which returns the smallest integer greater than or equal to $n/2$. $\lfloor (n-1)/2 \rfloor$ is the floor function, which returns the largest integer less than or equal to $(n-1)/2$.

$$S_{xy} = 0 * x_0 + 1 * x_1 + 2 * x_2 + 3 * x_3 + \cdots N * x_N \tag{11}$$

Equation 11 represents the sum of the product of each index n and the corresponding value x_n for a given number of

N indices. This is a weighted sum, where the weights are the indices, and it is used to compute the slope in linear regression.

Let S_{xx} be the sum of the squares of the indices, determining the variance of the indices.

$$S_{xx} = 0^2 + 1^2 + 2^2 + 3^2 + \cdots N^2 \tag{12}$$

Let S_y represents the total sum of the data points.

$$S_y = x_0 + x_1 + x_2 + x_3 + \cdots x_N \tag{13}$$

Let $\mathrm{S_x}$ be the sum of the indices from 0 to N.

$$S_x = 0 + 1 + 2 + 3 + \cdots N \tag{14}$$

Now, calculating the slope a of the best-fit line through the data points (n, x_n). The slope represents the rate of change of x_n concerning n.

$$a = \frac{S_{xy} - \frac{S_x * S_y}{N}}{S_{xx} - \frac{S_x * S_x}{N}} \tag{15}$$

Next, calculate the intercept b of the best-fit line, which is the expected value of x_n when $n = 0$.

$$b = S_y - \frac{a * S_x}{N} \tag{16}$$

Derive the $DSA[n]$ value for each index n. This involves finding the difference between the actual value of x_n and the linear regression line $an + b$.

$$DSA[n] = x_n - (a * n + b),$$
$$Where\ n = 0\ to\ N$$

Equation 17 represents $DSA[n]$, showing the deviation of the actual data points from the estimated linear trend. This helps in understanding the errors or variations in the data that are not explained by the linear model.

3.4. Reinforcement Learning Framework

DDPG algorithm [12] is utilized because it is stable, accurate, optimal, and efficient in solving problems with continuous action spaces. The DDPG algorithm balances performance with computational complexity well.

3.4.1. State Representation

The state representation includes:

- Dynamic Dither Angle, *DDA(θ_n):* The dynamic angular measurement of the gyroscope.
- Previous DDAngles (θ_{n-1}, θ_{n-2},...): Historical states of the gyroscope.
- Dither Symmetry Angle Values ($DSA[n]$): Measure of dither-induced errors.
- PRN Values ($PRN_A[n], PRN_B[n]$): Noise added to the dither angles.
- Error Metrics: Difference between expected and measured angles.

3.4.2. Action Representation

The actions involve adjusting the values for the dither angles $A[n]$ and $\mathrm{B}[n]$. Specifically, the RL agent takes the following actions.

Up	: row = max(row-1, 1)
Down	: row = min(row+1, nRows)
Left	: col = max(col-1, 1)
Right	: col = min(col+1, nCols)

These actions move the reference point up, down, left, or right [14] inside the grid's boundaries. Below is a thorough description of every action. The “Up” operation increments the location in the grid by one row. To ensure it stays within bounds, the position is changed to the maximum of the current row minus one and one. The “Down” operation moves the location one row lower. To keep it from going past the last row, the position is adjusted to the minimum of the current row plus one and the total number of rows (nRows).

The “Left” operation shifts the position of one column to the left, ensuring it stays within the first column by setting the position to the maximum of the current column minus one and one. The “Right” operation moves the position one column to the right, ensuring it does not go beyond the last column by setting the position to the minimum of the current column plus one and the total number of columns (nCols). These movements within the grid ensure that the position remains inside the grid's boundaries. The max and min functions are used to handle cases where a movement might cause the location to go outside the grid.

3.4.3. Reward Function Design

The reward function in reinforcement learning [15] plays a significant role as it guides the agent's decision process through immediate feedback on the agent's actions. Mathematically, the reward function describes how an agent must update its policy and value estimation.

$$R = \kappa \cdot (A - L) + \delta \tag{18}$$

Where κ is a scaling factor to adjust the overall influence of the combined metrics, A is the accuracy metric, defined as Equation 19.

$$A = \gamma . \frac{1}{n} \sum_{i=0}^{N-1} \left(\theta_i - \theta_{expected}\right)^2 \tag{19}$$

γ is a scaling factor. N is the number of components or observations. θ_i is the value observed at the *i-th* position, and $\theta_{expected}$ is the expected value for θ. L is the lock-in effect metric, defined as,

$$L = \alpha . L_{static} + \beta . L_{dynamic} \quad (20)$$

α and β are weighting factors. L_{static} and $L_{dynamic}$ are the static and dynamic lock-in effect metrics, respectively. δ is a constant offset value to ensure that the reward has a desirable range and is not zero nor negative.This makes the reward function one of the most important guiding mechanisms in the decision-making process for any agent through immediate feedback on its actions. Here are a few essential reasons:

Objective Definition: The reward function defines the task's objectives by assigning numerical values to different states and actions. It communicates what outcomes are desirable (positive rewards) and undesirable (negative rewards or zero rewards).

Feedback Loop: Rewards generate a feedback loop through which the agent learns by performing actions. A reward the agent receives based on an action puts the agent's ability to plan into practice in a way that maximizes cumulative rewards during the course of the action's lifetime.

Behavioural Guidance: Rewards shape the agent's behaviour, encouraging actions that result in higher rewards and discouraging those resulting in lower rewards. This goes a long way in directing the agent toward learning optimal strategies in pursuit of the desired objectives of the task.

3.4.4. Algorithm

```
→Initialize Environment and Parameters.
  a. Load the reward matrix $R$ from a .csv file.
  b. Define the grid world environment dimensions, the number of rows $n_{rows}$ and columns $n_{cols}$.
  c. Calculate the total number of states $n_{States} = n_{Rows} \times n_{Cols}$.
  d. Define the number of actions $n_{Actions}$ as 4.
→Identify Goal State
  a. Locate the goal state by finding the position with the maximum reward in the reward matrix $R$.
  b. Convert this position to the corresponding state index.
→Initialize Q-Table
  a. Create a Q-table with dimensions $n_{States} \times n_{Actions}$ Initially set to zero.
→Set Hyperparameters
  a. Learning rate $\alpha$.
  b. Discount factor $\gamma$.
  c. Exploration rate $\epsilon$.
  d. Maximum number of episodes maxEpisodes.
  e. Maximum steps per episode maxSteps.
→ Loop for training
  a. For each episode (from 1 to maxEpisodes).
      • Randomly initialize the starting state.
      • For each step within an episode (from 1 to maxSteps).
          ▪ Choose an action using the epsilon-greedy policy.
              • With probability $\epsilon$, select a random action.
              • Otherwise, select the action with the highest Q-value for the current state.
          ▪ Execute the chosen action and observe the next state and reward using the 'takeAction' function.
          ▪ Update the Q-value.
              • Update the Q-value for the current state-action pair using the Q-learning update rule.
              • $Q(state, action) \leftarrow Q(state, action) + \alpha(reward + \gamma * maxQ(nextState, a') - Q(state, action))$
              • Transition to the next state.
              • Check if the goal state is reached; if so, end the current episode.
        End for each step.
    End for each episode.
→Performance Tracking and Visualization.
  a. Track the total reward obtained in each episode.
  b. Plot the total reward per episode to visualize the learning progress.
```

3.4.5. Block Diagram

Figure 2 illustrates a reinforcement learning system designed to control a dither motor in an environment. The DDPG Reinforcement Learning algorithm is used to manage updates and generate the next action. The reward function evaluates performance metrics such as accuracy and lock-in effect, providing incentives for the RL algorithm.

The Epsilon-Greedy policy operates based on a probability distribution to balance exploration and exploitation [16]. Specifically, with a small probability ϵ, a random action is selected, and with the remaining probability $1-\epsilon$, the best action based on current knowledge is chosen. This selection process can be represented as a random action with probability ϵ and $\text{argmax}_a Q(s_t,a)$ with probability $1-\epsilon$. Here, a is the action taken at time t, $Q(s_t,a)$ represents the estimated value of taking action a in state s_t, and argmax_a denotes the action that maximizes the Q-value.

For the random action part, if all actions are equally likely, a uniform distribution can be used: Random action$\simeq$Uniform(A), where A is the set of all possible actions. These can be combined into a single equation using a random variable X, which is uniformly distributed between 0 and 1, for the selection process.

$$uniform(A), \quad if\ X < \epsilon \quad (21)$$

$$argmax_a Q(S_t, a), \quad if\ X \geq \epsilon, \quad (22)$$
$$where\ X \simeq uniform(0,1)$$

Equations 21 and 22 capture the essence of the epsilon-greedy policy by combining exploration and exploitation phases into a single decision-making process, as shown in Figure 3.

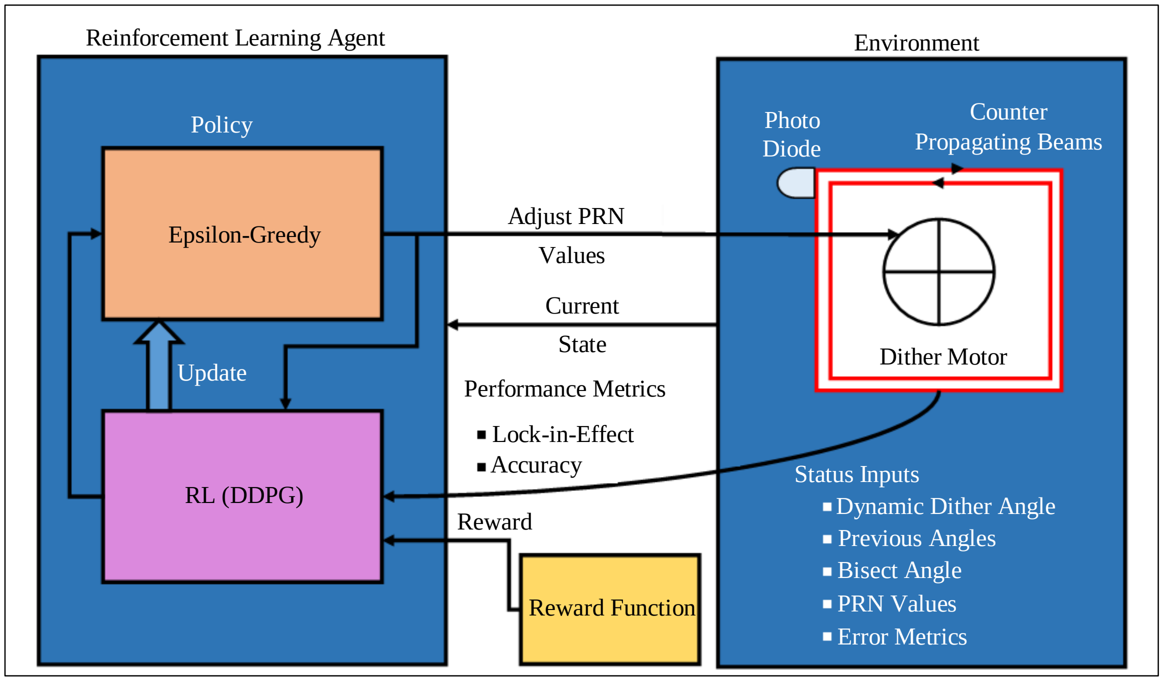

Fig. 2 DDPG based RL framework with RLG environment

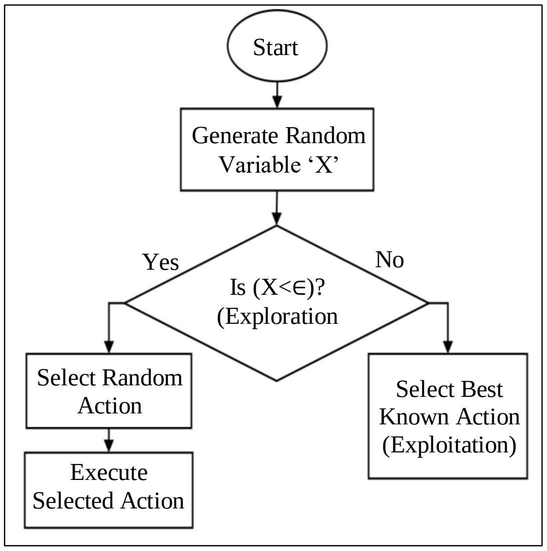

Fig. 3 Epsilon-Greedy policy flowchart for action selection

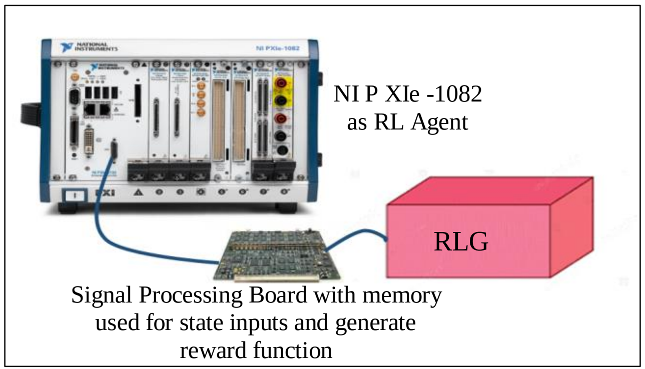
NI P XIe -1082
as RL Agent
RLG
Signal Processing Board with memory used for state inputs and generate reward function

Fig. 4 Experimental setup components and interactions

4. Experimental Setup

Figure 4 presents the test bench for the lock-in errors optimization of RLG through Reinforcement Learning. The chassis of NI PXIe-1082 is considered as the RL agent and will supervise tasks involved in data acquisition and control. A signal processing board interfaced with the chassis processes RLG signals and generates state inputs and rewards function calculation through the defined performance indexes. The RLG furnishes measurements of angular velocities, which are analyzed by the signal processing board. The RL agent uses that data to send control signals back to the RLG and change the dither amplitude and PRN values of the parameters, which helps to maintain the feedback loop for constant optimization.

Figure 5 shows a LabVIEW-based setup for a Reinforcement Learning agent controlling [6] an RLG system. It integrates Anaconda Python sessions within LabVIEW to execute the RL algorithm [17]. Performance metrics and current states are input into the Python session, where the RL agent processes them to generate rewards and determine actions using an epsilon-greedy policy. The agent adjusts the PRN values through three sets, optimizing the RLG's performance. The setup includes monitoring states and metrics, ensuring real-time feedback and continuous improvement. This configuration allows seamless interaction between LabVIEW and Python for efficient RL implementation.

4.1. Simulation Environment

Therefore, for the accurate simulation of the RLG system and to test the effectiveness of the method proposed, a comprehensive simulation environment is developed. The simulation environment embodies the critical components of the RLG and injects its dynamics from mechanical dithering, PRN injection, and the Reinforcement Learning (RL) framework. MATLAB/Simulink was utilized to model and simulate the RLG system. The implementation of the Reinforcement Learning algorithm was completed using Python libraries, such as TensorFlow, Keras, and Stable Baselines3, within the Anaconda IDE. The Gym library created a custom environment where the RLG system interfaces with the RL algorithm.

4.2. System Parameters and Configuration

The simulation was set up with realistic parameters to closely mimic the behavior of an actual RLG system. The key parameters are described below. Table 1 outlines the key parameters for the RLG, which is integral to the system's precision and accuracy. The dither frequency ensures the stability of the gyroscope's output. The Dither Amplitude, dynamically adjusted through Pseudo-Random Noise (PRN) values, allows real-time modulation to optimize performance under varying conditions. The laser wavelength is known for its stability and coherence. The cavity length influences the scale factor and sensitivity of the RLG.

Table 1. RLG parameters

Parameter	Value
Dither Frequency	357.8 Hz
Dither Amplitude	Adjusted dynamically through PRN values
Laser Wavelength	632.8 nm
Cavity Length	32cm

Table 2 specifies the parameters related to the Pseudo-Random Noise (PRN) utilized in the system. The frequency range of the PRN provides a broad spectrum that enhances the robustness of the dither signal.

The amplitude range ensures sufficient variability to prevent signal degradation while maintaining control. The update interval for the PRN values is set at every 10 milliseconds, allowing for rapid adjustments and fine-tuning of the system's performance.

Table 2. PRN parameters

Parameter	Value
Frequency range	0.1 to 10Hz
Amplitude range	0.01 to 0.1 radians
Update Interval	Every 10ms

Table 3 lists the Reinforcement Learning (RL) parameters employed in the systems optimization. The DDPG continuously creates action spaces that balance exploration and exploitation effectively. The learning rate dictates the speed of learning and convergence of the RL agent. The discount factor reflects the agent's preference for long-term rewards over immediate gains.

The policy network is described as a DDPG agent with fewer episodes and steps, tailored for efficient learning without extensive computational demands. Finally, the training episodes total 1000, providing a substantial dataset for the agent to learn and refine its policy.

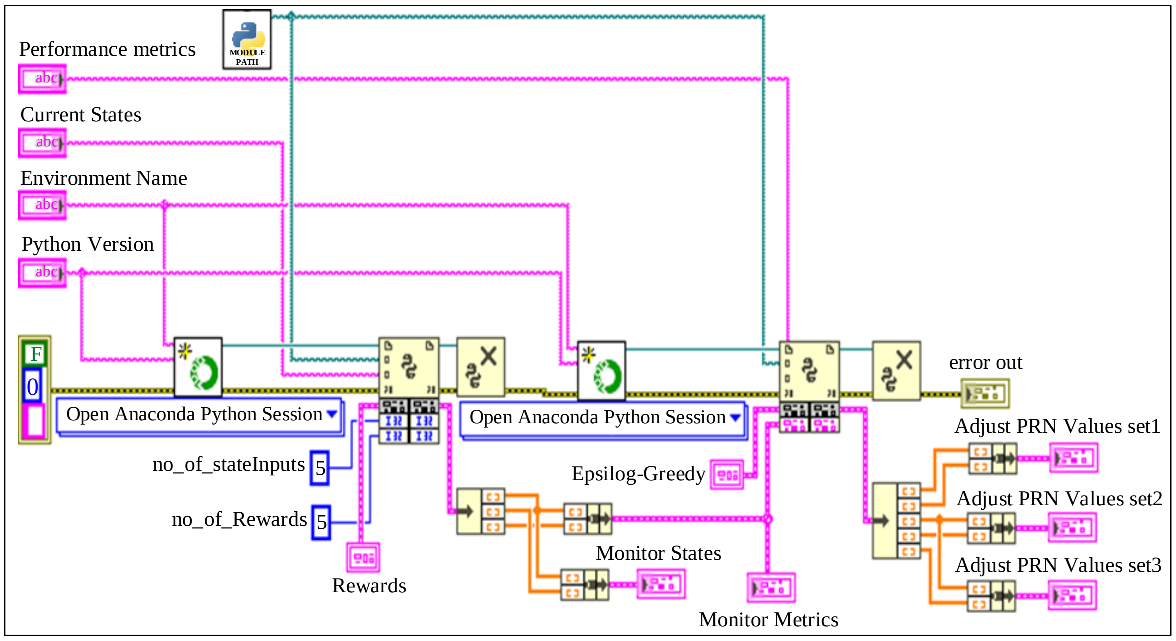

Fig. 5 Block diagram of LabVIEW based reinforcement learning agent

Table 3. RL parameters

Parameter	Value
Algorithm	DDPG
Learning Rate	0.1 sec
Discount Factor(γ)	0.9
Policy Network	DDPG agent with fewer episodes and steps
Training Episodes	1000
Steps per Episode	100

5. Experimental Results

Table 4 illustrates the effects of added noise on the values A[n] and B[n], as well as the differences A[n]-B[n], compared to a scenario without noise. Using Equation 6, the Dynamic Dither Angle (DDA) is plotted and displayed in Figure 6. Similarly, using Equation 17, the Dither Symmetry Angle (DSA) is plotted and shown in Figure 7. These plots represent the statistical signal analysis of adding noise to the ring dither.

Table 4. Comparison of measurements with and without added noise

Sample Number (n)	Added noise (pulses)			Without noise (pulses)		
	A[n]	B[n]	A[n]-B[n]	A[n]	B[n]	A[n]-B[n]
0	214	214	0	217	217	0
1	214	213	1	217	217	0
2	212	211	1	217	217	0
3	210	210	0	217	217	0
4	210	209	1	217	217	0
5	208	207	1	217	217	0
6	206	206	0	217	217	0
.	-	-	-	-	-	-
.	-	-	-	-	-	-
4095	197	198	-1	217	217	0

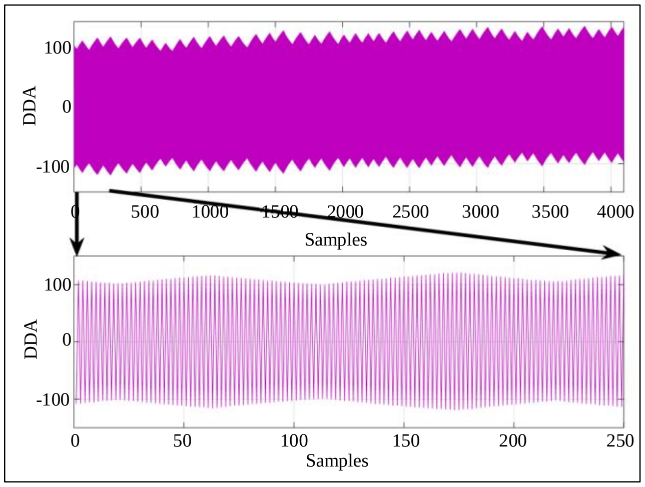

Fig. 6 Dynamic Dither Angle (DDA) over time

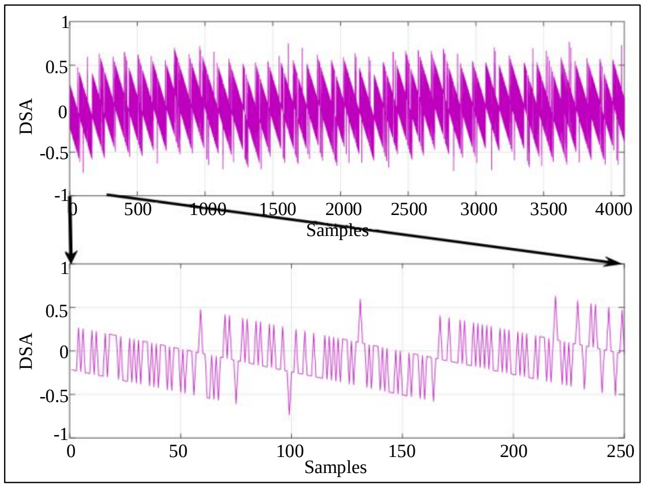

Fig. 7 Dither Symmetry Angle (DSA) over time

5.1. Data Collection and Processing

To evaluate the performance of the proposed method, data was collected from the simulation environment over multiple episodes. The types of data collected are listed in Table 5.

Table 5. Data collection and processing

Data Type	Description
DDA(θ_n)	Measured at each step
Previous Angle	Historical states of the RLG
DSA	Calculated based on the method described
PRN Values	Adjustments made by the RL agent
Error Metrics	Difference between the expected and measured angles.

The data was then processed to generate performance metrics, including lock-in occurrence and measurement accuracy. These metrics were used to assess the effectiveness of the proposed method.

5.2. Performance Metrics

The following metrics (see Table 6) were used to evaluate the performance of the proposed methodology.

Tabe 6. Performance metrics

Metric	Description
Lock-in Occurrence	Frequency and duration of static and dynamic lock-in events.
Measurement Accuracy	Mean Squared Error (MSE) between the expected and measured angles.
Convergence Rate	Number of episodes required for the RL agent to converge to an optimal policy.
Computational Efficiency	Time taken for training and inference.

5.3. Experimental Procedure

The experimental procedure involved the steps outlined in Table 7.

Table 7. Experimental procedure

Step	Description
Initialization	Initialize the RLG system parameters, PRN values, and the RL agent.
Training	Train the RL agent using the DDPG algorithm in the simulated environment. The agent adjusts the PRN values based on the state inputs and receives rewards based on the performance metrics.
Evaluation	After training, evaluate the RL agent's performance by running multiple test episodes. Collect data on lock-in occurrence and measurement accuracy.
Analysis	Analyze the collected data to determine the effectiveness of the proposed method in reducing lock-in effects and improving accuracy.

5.4. Validation

To validate the simulation results, the proposed method was compared with traditional dithering methods and static PRN injection. The details of the validation process are provided in Table 8.

Table 8. Validation

Step	Description
Baseline Comparison	Comparing the results with a baseline where no PRN was injected and only mechanical dithering was used.
Statistical Analysis	Performing statistical tests to determine the significance of the improvements observed.

5.5. Results and Analysis

Figure 8 illustrates how an algorithm learns over 1000 episodes with 40% of randomized data. The x-axis represents the episodes, while the y-axis shows the total reward. The plot displays the algorithm's performance for each episode, with noticeable variation. Despite these fluctuations, the rewards tend to center around a mean value, indicating that the system has stabilized its performance after the initial learning phase. The occasional sharp drops, such as the significant one around episode 430, may indicate underperforming episodes or exploration phases.

Figure 9 shows the learning progress of an algorithm over 1000 episodes for 13% of randomized data, with a 13% error on randomized data (sample size 4096*3). The rewards are generally stable around a mean value, but there are notable dips at several points, suggesting occasional poor performance or periods of exploration.

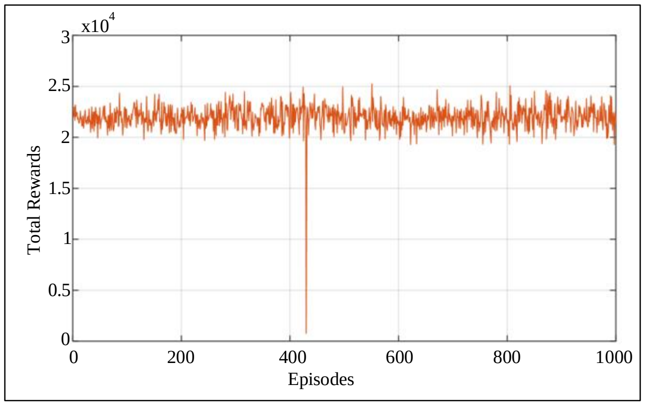

Fig. 8 Learning process showing error (spike) in 40% of randomized data (sample size = 4096)

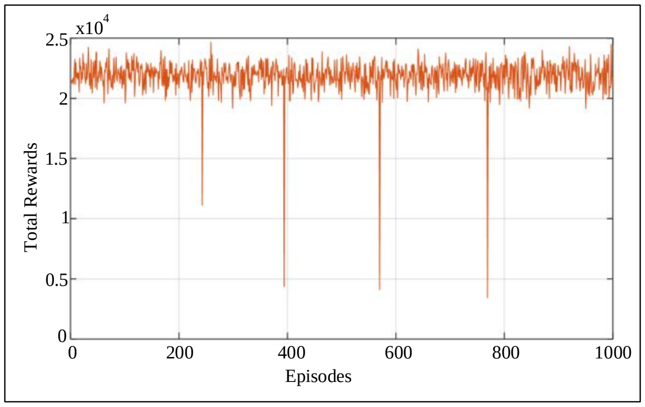

Fig. 9 Learning process showing error (spikes) in 13% randomized data (sample size=4096*3)

Figure 10 suggests that the sample size of 40,960 is large enough to estimate a population parameter with a 3% margin of error at a specific confidence level. This small margin of error indicates a high degree of precision in estimating the true population parameter, as larger sample sizes typically result in smaller margins of error. The cumulative reward progress over episodes is plotted on a graph that oscillates around a particular mean, reflecting improved performance in the learning process.

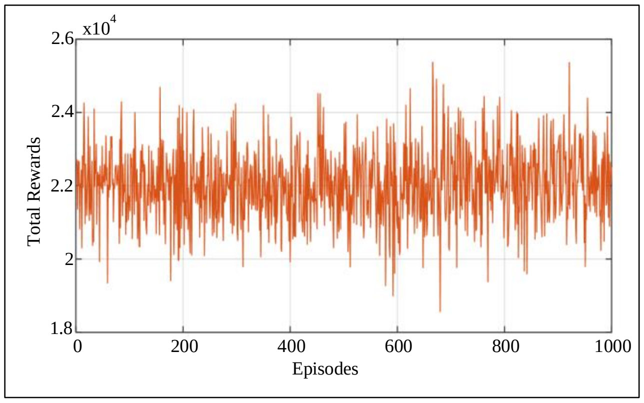

Fig. 10 Learning process showing error (spikes) of 3% for randomized data (sample size=4096*10)

6. Conclusion

This research introduces a novel method for enhancing the performance of Ring Laser Gyroscopes (RLGs) by integrating Statistical Signal Processing (SSP) and Reinforcement Learning (RL). It tackles the significant issue of static and dynamic lock-in effects that reduce the accuracy of RLGs, especially in low angular rate conditions.

By introducing the concepts of Dynamic Dither Angle (DDA) and Dither Symmetry Angle (DSA), a framework is developed to effectively capture the cumulative and balanced effects of dither pulses. The DDA and DSA calculations provide detailed insights into the error patterns induced by dithering, allowing for precise adjustments to reduce lock-in effects.

Using the Deep Deterministic Policy Gradient (DDPG) algorithm further strengthens this approach. The DDPG-based RL framework dynamically optimizes the Pseudo-Random Noise (PRN) values, effectively minimizing both static and dynamic lock-in occurrences. This dynamic adjustment is achieved through the interaction of the RL agent with the RLG system, continuously refining the dither strategy based on feedback from performance metrics such as lock-in occurrence and angular measurement accuracy.

Extensive simulations validated the efficacy of the proposed methods, demonstrating significant reductions in lock-in effects and notable improvements in gyroscope performance. The integration of SSP and RL offers a robust and efficient solution, advancing RLG technology and making it more reliable for precise navigation applications.

In conclusion, this research successfully combines Statistical Signal Processing with Reinforcement Learning to address a critical challenge in RLG performance. The proposed approach not only reduces lock-in effects but also enhances the overall accuracy and reliability of RLGs, paving the way for their more effective use in advanced navigation systems. Future work could explore further optimizations and real-world implementations to solidify these findings and extend their applicability.

References

[1] Woo-Seok Choi, “Sagnac Effect Compensations and Locked States in a Ring Laser Gyroscope,” *Sensors*, vol. 23, no. 3, pp. 1-10, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[2] Zhenfang Fan et al., “Random Walk Reduction in Dithered Ring Laser Gyroscope,” *Optica Express*, vol. 31, no. 23, pp. 37959-37967, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[3] Angela D.V. Di Virgilio et al., “Noise Level of a Ring Laser Gyroscope in the Femto-Rad/s Range,” *Physical Review Letters*, vol. 133, no. 1, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[4] Igor Kudelin, Srikanth Sugavanam, and Maria Chernysheva, “Rotation Active Sensors Based on Ultrafast Fibre Lasers,” *Sensors*, vol. 21, no. 10, pp. 1-30, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Danilo Bzdok, Naomi Altman, and Martin Krzywinski, “Statistics versus Machine Learning,” *Nature Methods*, vol. 15, pp. 233-234, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[6] Alexey Kokhanovskiy et al., “Machine Learning Methods for Control of Fibre Lasers with Double Gain Nonlinear Loop Mirror,” *Scientific Reports*, vol. 9, pp. 1-7, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[7] Luca Pasqualini, and Maurizio Parton, “Pseudo Random Number Generation: a Reinforcement Learning Approach,” *Procedia Computer Science*, vol. 170, pp. 1122-1127, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[8] Qiang Du et al., “Experimental Beam Combining Stabilization Using Machine Learning Trained While Phases Drift,” *Optics Express*, vol. 30, no. 8, pp. 12639-12653, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Yang Jianqiang, Jia Xueqing, and Yuan Baolun, “Digital Detection and Control System for Ring Laser Gyro,” *IEEE 2010 2nd International Conference on Advanced Computer Control*, Shenyang, China, pp. 216-220, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[10] Jun Weng et al., “Optimization of Ring Laser Gyroscope Bias Compensation Algorithm in Variable Temperature Environment,” *Sensors*, vol. 20, no. 2, pp. 1-13, 2020. [CrossRef] [Google Scholar] [Publisher Link].

[11] Guo Wei et al., “Application of Least Squares-Support Vector Machine in System-Level Temperature Compensation of Ring Laser Gyroscope,” *Measurement*, vol. 44, no. 10, pp. 1898-1903, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[12] Ebrahim Hamid Sumiea et al., “Deep Deterministic Policy Gradient Algorithm: A Systematic Review,” *Heliyon*, vol. 10, no. 9, pp. 1-26, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[13] Frederick Aronowitz, *Fundamentals of the Ring Laser Gyro*, Optical gyros and their Application, North Atlantic Treaty Organization & Research and Technology Organization, France, pp. 3.1-3.15, 1999. [Google Scholar] [Publisher Link]

[14] Zelin Wan et al., “Decision Theory-Guided Deep Reinforcement Learning for Fast Learning,” *arXiv*, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[15] Ahmed Abouelazm, Jonas Michel, and J. Marius Zoellner, “A Review of Reward Functions for Reinforcement Learning in the context of Autonomous Driving,” *arXiv*, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[16] Michel Tokic, “Adaptive ε-Greedy Exploration in Reinforcement Learning Based on Value Differences,” *Advances in Artificial Intelligence*, pp. 203-210, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[17] Ankur Kumar, and Mayank Goswami, “Performance Comparison of Instrument Automation Pipelines Using Different Programming Languages,” *Scientific Reports*, vol. 13, pp. 1-14, 2023. [CrossRef] [Google Scholar] [Publisher Link]