School of Industrial Engineering

Course: Non-linear systems

Control of the Monza System with non-linear methods

Author: Martín Loring Bueno

Degree: Master's Degree in Automation and Robotics

January 27, 2025

Contents

List of Figures

1 Introduction

Monza is a Spanish mechanical arcade game from 1979 in which the player inserts a coin into the top slot and directs it using a steering wheel that controls the inclination of a tilted circuit. The goal is to guide the coin to the bottom exit without it falling off the sides. The circuit, designed with parabolas and alternating rails, simulates a racetrack, and the objective is to recover the coin upon successfully completing the course.

In this first section, we will focus exclusively on deriving the equations governing the system's dynamics. This dynamic is based on the motion of the coin through a parabola, its subsequent fall, and reception. Thus, equations will be derived for each of these different physical states in which the coin can find itself. Additionally, before starting, emphasis must be placed on the reference system to be used.

2 Derivation of the Equations

2.1 Reference Systems

Due to the complex system that constitutes the game, special attention must be paid to the reference system used when forming the coin's kinematic equations. In the game, the coin rolls along the parabolic paths formed by the different levels it falls through. There is a reference system where the x-axis is aligned with the middle of the parabola, and the y-axis is centered in the middle of the game. This serves as a reference to measure the inclination angle of the parabola along which the coin falls. This reference system, hereafter referred to as $x_{parabola}$ and $y_{parabola}$, can be seen in Figure 1, taken from the course assignment notes:

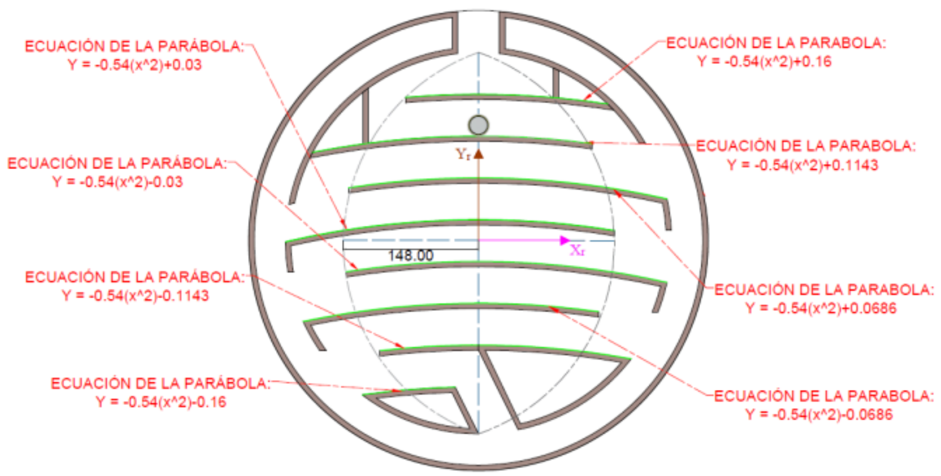


Figure 1: Diagram of the game showing the parabola's reference system

In this way, if we take another reference system located at the very center of the coin, and move it so that its x-axis is parallel to the track along which the coin rolls, an angle α will be formed between this system and the previously introduced one, as seen in figure 2. As can be observed, a 5-peseta Spanish coin is used in this image, which was the coin commonly used in this arcade game.

Figure 2: Diagram of the coin's movement along the track and the reference systems

At this point, it is trivial to deduce that this angle α is defined by the slope of the parabola in which the coin is located, as seen in equations 1 and 2, where the slope of the parabola is its derivative with respect to x, where x refers to the previously mentioned reference system $x_{parabola}$. The equations defining the parabolas are given in the project statement, as seen in figure 1, and are of the kind $y = -0.54x^2 + a$.

$$Slope = \frac{dy}{dx} = \frac{-0.54x^2 + a}{dx} = -1.08x_{parabola} \tag{1}$$

$$\alpha = \arctan(\text{slope}) = \arctan(-1.08x_{parabola}) \tag{2}$$

However, we cannot work solely with these two reference systems, since there is another motion that moves the coin: the angle introduced through the steering wheel that controls the game. Since this angle, which we will call θ, makes both previously discussed reference systems move, we must create another reference system that is unaffected by the game's motion. This system will serve as our reference, and its axes will be $x_{reference}$ and $y_{reference}$, as seen in Figure 3.

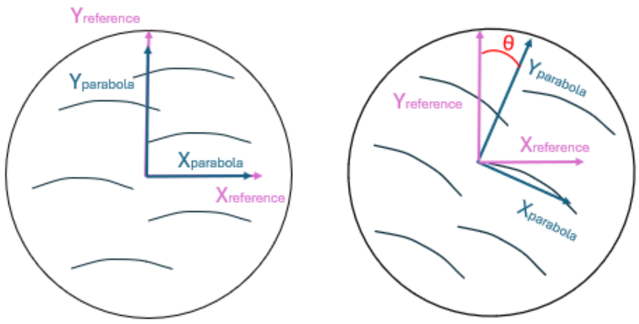


Figure 3: Diagram of the game showing the reference system with the angle that rotates the game

Once the reference systems have been defined and a non-inertial reference system has been chosen, since its position does not change, Newton's laws can be applied.

2.2 Rolling along the tracks

The first motion the coin undergoes is the fall along each of the tracks in the game. This movement can be simplified as the fall of a round object along an inclined plane, as shown in Figure 4, where the angle of inclination is the sum of the two previously observed angles: α and θ.

In this movement, the force of gravity will appear, with its x and y components, and the frictional force, which defines the motion of the coin according to equation 3.

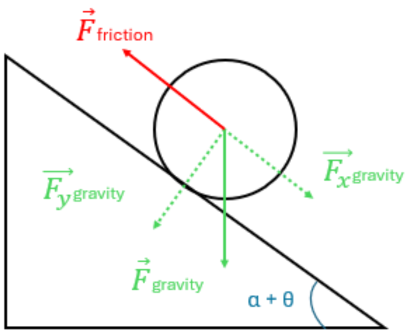


Figure 4: Coin's movement along the track with the kinematics of an inclined plane

$$F_{x_{gravity}} - F_{friction} = mass_{coin} \cdot acceleration_{coin} \tag{3}$$

In addition to these forces, it must be considered that the coin will roll along the parabola, so it will have angular acceleration and velocity. Therefore, equations 4 and 5 are added, where R is the radius of the coin and I_C is its moment of inertia.

$$F_{friction} \cdot R = I_c \cdot acceleration_{angular} \tag{4}$$

$$acceleration = acceleration_{angular} \cdot R \tag{5}$$

Thus, it is possible to substitute in equation 3 and reach the following deduction:

$$F_{x_{gravity}} - F_{friction} = mass_{coin} \cdot aceleracion_{coin}$$

$$F_{x_{gravity}} \cdot sin(\alpha + \theta) - F_{friction} = mass_{coin} \cdot acceleration_{coin}$$

$$g \cdot m \cdot sin(\alpha + \theta) - \frac{I_c \cdot accelerationn_{angular}}{R} = m \cdot a_{coin}$$

$$g \cdot m \cdot sin(\alpha + \theta) - \frac{I_c \cdot a_{coin}}{R^2} = m \cdot a_{coin}$$

And returning to equation 2, the angle α was expressed as:

$$g \cdot m \cdot sin\{\arctan(-1.08x_{parabola}) + \theta\} - \frac{I_c \cdot a_{coin}}{R^2} = m \cdot a_{coin}$$

Thus, our equation for controlling the coin while it rolls along the tracks will be equation 6, where, depending on the input angle θ and the coin's position, it will roll with an acceleration a_{coin}.

$$g \cdot sin\{\arctan(-1.08x_{parabola}) + \theta\} - \frac{I_c \cdot a_{coin}}{m \cdot R^2} = a_{coin} \tag{6}$$

In addition to this equation, we have the equations of the parabolas, which relate $x_{parabola}$ with $y_{parabola}$, as shown earlier in Figure 1, and these are in the form seen in equation 7, where a is a value that depends on the level we are at.

$$y_{parabola} = -0.54x^2_{parabola} + a \tag{7}$$

When simulating this equation in Simulink to control the coin, the acceleration can be integrated, which will give us the velocity and position. However, it must be noted that this position refers to the inertial reference system, while in the equation, there is another parameter, $x_{parabola}$, whose position is taken concerning the reference system of the parabola. To work with both parameters, a rotation must be made that relates both reference systems, where the phase angle is the angle θ. In equation 8, the equation that relates $x_{parabola}$ with $x_{reference}$ and $y_{reference}$ is shown. After introducing this variable $y_{reference}$, we must introduce another one that solves this, using the rotation equation that relates $y_{reference}$ with $x_{parabola}$ and $y_{parabola}$, as seen in equation 9.

$$x_{parabola} = x_{reference} \cdot cos(\theta) + y_{reference} \cdot sin(\theta) \tag{8}$$

$$y_{reference} = x_{parabola} \cdot sin(\theta) + y_{parabola} \cdot cos(\theta) \tag{9}$$

Thus, with equations 6, 7, 8, and 9, the control of the coin, while it rolls through the different tracks, will be performed.

Finally, it is worth noting that to derive these equations, it has been assumed that the control will be done with small angle changes, i.e., the system will rotate slowly with small angle changes. If the angles were to change with high speeds and large angles, forces could appear that cause the coin to be launched and lose contact with the track. Therefore, it must be considered that the coin only loses contact with the track when it falls downward between levels.

2.3 Free fall between tracks

The next motion experienced by the coin is a free fall after reaching the end of one of the tracks. Figure 5 shows the components defining the kinematics of this motion, where the coin falls with an initial velocity, and the only force affecting the motion is gravity, assuming there is no air resistance.

The coin reaches the end of the track with its reference system offset by an angle α due to the parabola's inclination, and an angle θ due to the introduced inclination angle. Its velocity will therefore decompose into x and y components, as shown in equations 10 and 11.

$$V_Y o = V_o \cdot \sin(\alpha + \theta) \tag{10}$$

$$V_X o = V_o \cdot \cos(\alpha + \theta) \tag{11}$$

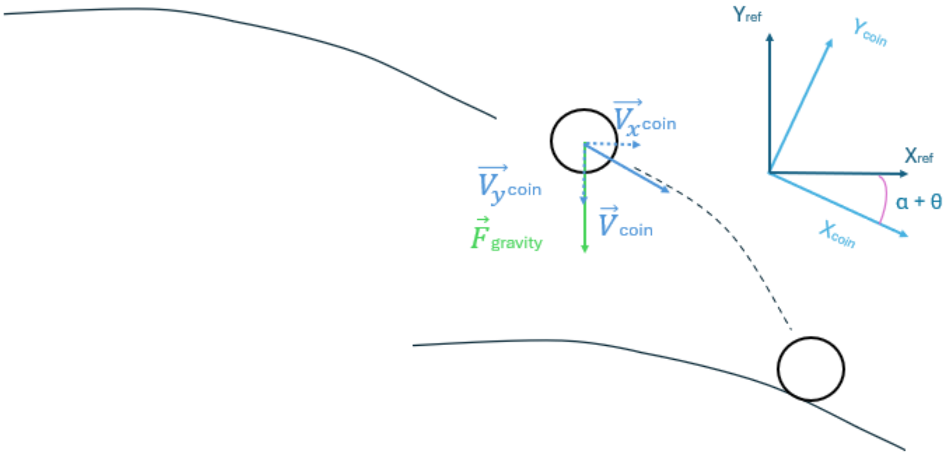


Figure 5: Kinematics of the coin falling from one track to another

Thus, the coin will undergo uniformly accelerated rectilinear motion along the y_{ref} axis and uniform rectilinear motion along the x_{ref} axis. The position and velocity equations for this motion are shown in equations 12, 13, 14, and 15.

$$y(t) = y_o + V_Y o \cdot t + \frac{1}{2} \cdot g \cdot t^2 \tag{12}$$

$$x(t) = x_o + V_X o \cdot t \tag{13}$$

$$V_y(t) = V_Y o + g \cdot t \tag{14}$$

$$V_x(t) = V_X o \tag{15}$$

The initial velocities and positions derived from the previous track motion are known from these equations. This allows the coin's positions and velocities to be precisely described during the fall. Once the coin lands on the lower track, the equations from the previous section will be reused, considering the velocity and position at which it lands according to these equations.

2.4 Reception in the lower track

Concerning landing on the lower rail after falling, it will be assumed that the coin does not bounce; instead, when it touches the lower rail, it begins its motion along the rail governed by the control equations discussed in Section 2.2. In this motion, the coin will reach the lower rail with the x- and y-position and the x-velocity given by the falling equations. The y-velocity will not be considered, as it will be halted by the rail surface upon impact.

2.5 Summary of the equations

Finally, the equations used for the control of the coin can be divided into:

Rolling along the tracks:

$$g \cdot sin\{\arctan(-1.08x_{parabola}) + \theta\} - \frac{I_c \cdot a_{coin}}{m \cdot R^2} = a_{coin}$$

$$y_{parabola} = -0.54x_{parabola}^2 + a$$

$$x_{parabola} = x_{reference} \cdot cos(\theta) + y_{reference} \cdot sin(\theta)$$

$$y_{reference} = x_{parabola} \cdot sin(\theta) + y_{parabola} \cdot cos(\theta)$$

Free fall between tracks:

$$y(t) = y_o + V_Y o \cdot t + \frac{1}{2} \cdot g \cdot t^2$$

$$x(t) = x_o + V_X o \cdot t$$

$$V_y(t) = V_Y o + g \cdot t$$

$$V_x(t) = V_X o$$

Along with the structure of the game, where the equations for each parabola, their distances, and the distances between them are provided, these equations allow the control of the game, with the angle θ being the input for this control.

2.6 Model testing

This text until now is the same as the one delivered for the first task of this work, deducing the dynamic equations of the model. After this, a model of the Monza game was given by the professor to test our own equations.

With this model, seen in Figure 6, we can test our model, seen in Figures 7a and 7b, for the equations of the coin rolling down the track and the free fall between tracks.

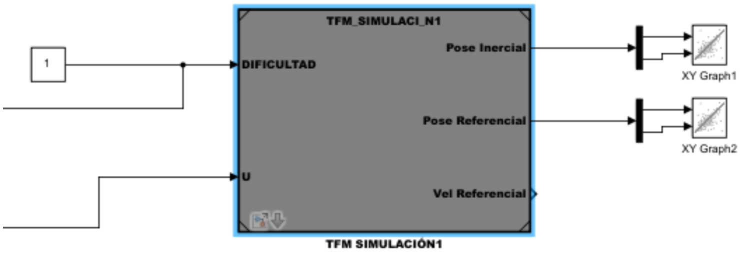
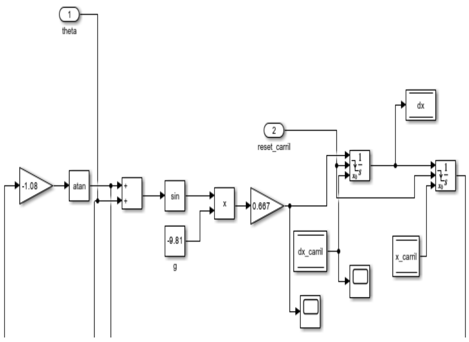
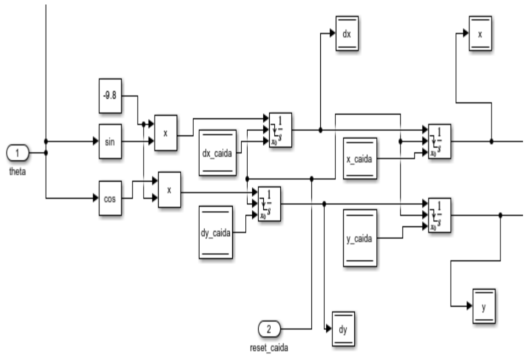


Figure 6: Monza model simulator given by the professor



(a) Coin rolling down the tracks



(b) Coin free fall

Figure 7: Monza coin dynamic equations in Simulink

During this test, it is clear that our equations do not work well with the simulations, because some mistakes were made. Firstly, in the equations for the free fall of the coin between tracks, it was supposed that the coin undergoes a uniformly accelerated rectilinear motion along the y_{ref} axis and uniform rectilinear motion along the x_{ref}, with no action of the angles θ and α. However, after a later study of the problem (Figure 5), it seems that the angle α does not affect the coin during the fall, since there is no angle with the track, but the angle θ can affect the movement since the game can be moved during this time. For this reason, a better explanation of this problem would be a uniformly accelerated rectilinear motion in both axes, governed by equations 16 and 17, in which the angle θ can be changed.

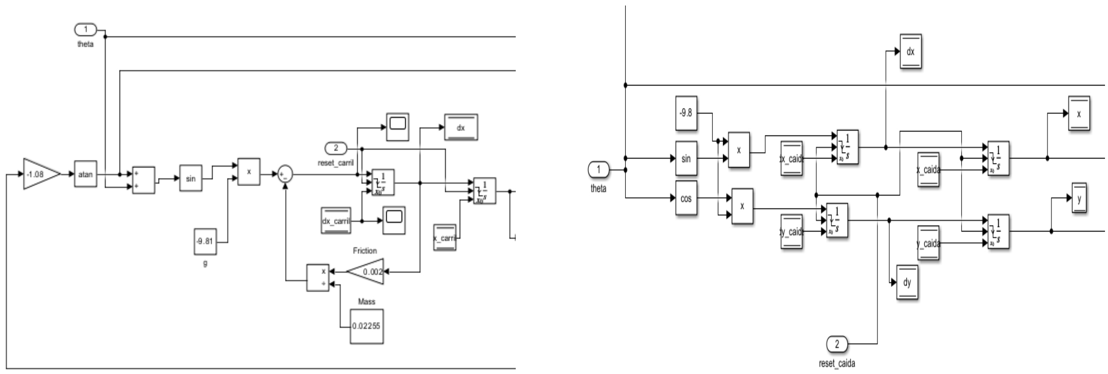
$$x_{ref} = x_o + V_Y o \cdot t + \frac{1}{2} \cdot cos(\theta) \cdot g \cdot t^2 \tag{16}$$

$$y_{ref} = y_o + V_Y o \cdot t + \frac{1}{2} \cdot sin(\theta) \cdot g \cdot t^2 \tag{17}$$

Along with this change, the movement in the tracks does not work properly either. For this reason, some physics books have been checked to analyzed how the frictional force can be calculated. Finally, it is found that the frictional force is proportional to the velocity of the coin multiplied by a viscosity coefficient [1]. With this change, equation 6 has been change to the following equation 18.

$$g \cdot sin\{\arctan(-1.08x_{parabola}) + \theta\} - \frac{\mu \cdot v_{coin}}{m} = a_{coin} \tag{18}$$

To check whether our model behaved like the one given, an angle of -0.2 radians was given to both models, and they both reacted similarly. For this reason, this model was kept to the control, and if anything needed to be changed, it would be done later. The final Simulink block diagram used is the one seen in Figure 8. Along with these equations, it is also important to remember the equations needed to rotate from one reference system to the other (equations 8 and 9), which have been implemented in Simulink as a block function.



(a) Coin rolling down the tracks

(b) Coin free fall

Figure 8: Improved Monza coin dynamic equations in Simulink block diagram

3 Model Identification

After deducing the equations that control the movement of the coin, the next step is to identify the model parameters. For this, it is important to notice that the equations that define the Monza game, like the level and difficulties, and equations of the tracks, are given by the professor, along with the interface to simulate in MATLAB.

So, this given, the only parameters that we need, seeing the equations previously discussed, are the gravity, the coefficient of viscosity and the mass of the coin.

For the force of gravity, a good estimation commonly used is 9.8 m/s^2, so this will be the one used. For the mass of the coin, a quick Google search will tell us that a Spanish 5 *pesetas* coin weights around 22.50 grams, so this will be the weight used, The coefficient of viscosity is a little more complicated, but a good estimated searching online is to use around 0.02 $\frac{m}{s}$.

4 Control alternatives

With all this in mind, the next step is to try to control this non-linear model, trying to make the coin fall from the first track to the last one, and thus winning the game.

As seen in the previous equations, this system is clearly non-linear, so the classical PID controllers seen during the degree are not applicable. For this control, one of the non-linear controllers studied in this and other courses must be used. Some of these methods are **fuzzy**, **reinforcement learning**, **neural networks** and **adaptive control**.

4.1 Fuzzy control

Fuzzy logic operates by defining rules based on linguistic terms like “near”, “fas”, or “slow”, rather than exact numerical values. It allows the system to interpret inputs such as the coin’s position and speed in a flexible way, producing smooth and adaptive outputs, like adjusting the rails. This makes fuzzy logic effective for controlling systems with complex, nonlinear behaviors, offering a simple yet powerful method for achieving intelligent and responsive control. Controlling the Monza game with fuzzy logic is ideal because this approach uses a system that can handle imprecise or uncertain information, similar to how humans make decisions.

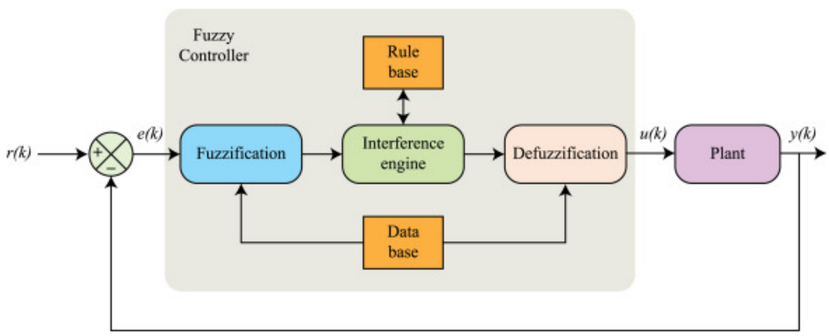


Figure 9: Fuzzy control block diagram sketch [2]

4.2 Reinforcement learning

Reinforcement learning works by training an agent to take actions that maximize cumulative rewards over time, without needing prior knowledge of the system’s dynamics. The agent explores different strategies, learning from its successes and failures, to improve its performance. This makes reinforcement learning especially effective for managing complex, dynamic, and uncertain scenarios, providing intelligent and adaptive control in tasks like guiding the coin through the rails in Monza.

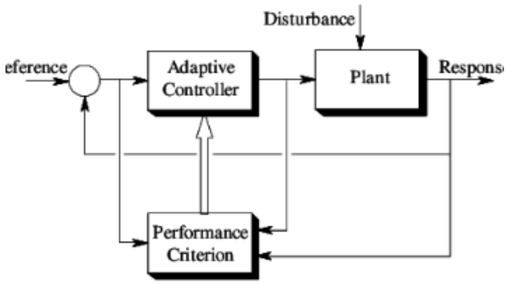


Figure 10: Reinforcement learning control block diagram sketch [3]

4.3 Neural networks

Using neural networks to control the Monza game could be a good approach because these systems, modeled after the human brain, are capable of learning and generalizing patterns from data. Neural networks are composed of layers of interconnected nodes (neurons) that process inputs—such as the coin's position and speed—and generate outputs, like the necessary rail adjustments. Through training on example scenarios, they can approximate complex relationships without the need for predefined rules or explicit programming. Their ability to capture nonlinear dynamics and adapt to changes makes neural networks a versatile and powerful solution for intelligent control in the dynamic and unpredictable environment of Monza.

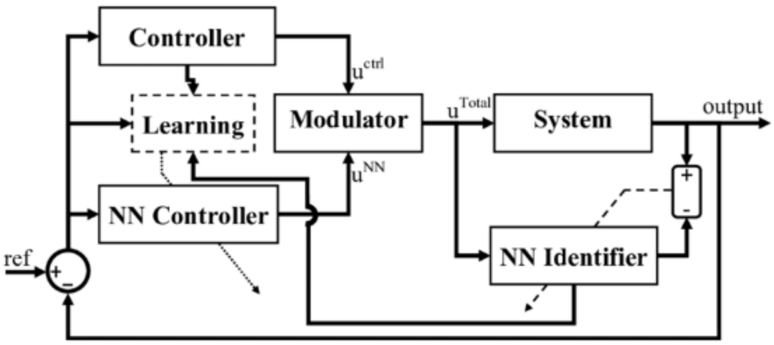


Figure 11: Neural Networks control block diagram sketch [4]

4.4 Adaptive control

Neural Networks, Reinforcement learning and fuzzy logic have already been seen in other courses during this Masters' Degree. However, in this course, Non-linear systems, another method to approach a problem of this complexity has been given: adaptive control.
An adaptive controller is one that has adjustable parameters and a mechanism to adjust those parameters [5]. This approach ensures that the system can maintain desired performance even

when there are uncertainties or variations in the system's dynamics or external disturbances. In case of the control of the Monza game, it would be highly effective, because this approach allows the system to adjust its behavior in real time based on changing conditions or uncertainties. Adaptive control continuously monitors the coin's position, speed, and other factors, updating the control strategy dynamically to ensure optimal performance. This is particularly useful in Monza, where the system must respond to nonlinear dynamics and variations in the coin's movement.

On top of this, adaptive control could be implemented easily in Simulink with transfer functions, while control with other methods such as reinforcement learning could be harder. For these reasons, and with the advice of the professor who recommended using adaptive control, an adaptive control will be implemented.

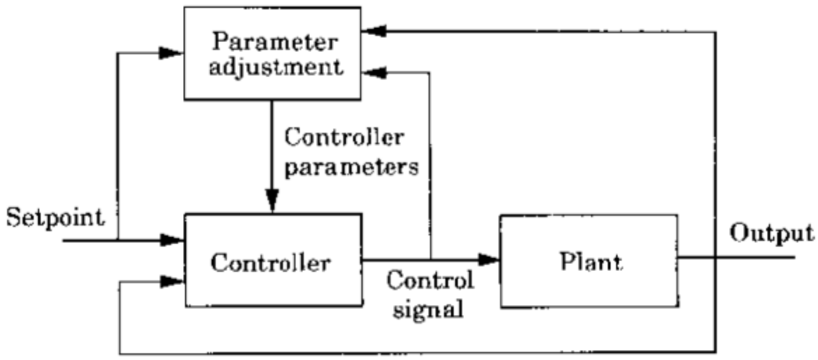


Figure 12: Adaptive control block diagram. Adaptive control will be the one implemented [6]

5 Controllers Implementations

To be able to control this system, we must go back to the equation we are trying to control, which is equation 18. To control this equation, we must deduce the transfer function that defines the system from it. To simplify the controller design, the nonlinear equations are linearized. This involves approximating the equations around an equilibrium point ($x_0 = 0$, $\theta_0 = 0$), which is done in equation 19.

$$\Delta\ddot{x} = \frac{\partial(g \cdot \sin(\alpha(x) + \theta))}{\partial x}\Delta x + \frac{\partial(g \cdot \sin(\alpha(x) + \theta))}{\partial \theta}\Delta\theta - \frac{\partial\left(\frac{\mu \cdot \dot{x}}{m}\right)}{\partial \dot{x}}\Delta\dot{x} \tag{19}$$

Which can be expressed as:

$$\Delta\ddot{x} = A\Delta x + B\Delta\theta - C\dot{x}$$

To analyze the system in the frequency domain, a Laplace transform is applied to the linearized equation. This produces a transfer function relating displacement δx to the angle $\delta\theta$ [7]:

$$G(s) = \frac{\Delta x}{\Delta\theta} = \frac{B}{s^2 + Cs - A}$$

This transfer function indicates that the system behaves as a second-order system, which is typical for mechanical systems with inertia and damping.

With known values for the mass, viscosity coefficient and gravity, the values for A, B and C can be calculated:

$$A = 10.584; B = -9.8; C = \frac{\mu}{m} = 0.89$$

From this, the transfer function of the system can be expressed as:

$$G(s) = \frac{\Delta x}{\Delta\theta} = \frac{B}{s^2 + Cs - A} = \frac{-9.8}{s^2 + 0.89s - 10.584}$$

Once it is clear that the system behaves as a second-order system, the next thing that is needed is to choose a reference model for the control of the system [8]. For this, the parameters from equation 20 must be decided. For the system to behave as an underdamped system, a damping ratio of 0.6 has been chosen. with a natural frequency of 3 rad/s, leaving us with our reference model seen in equation 21.

$$H(s) = \frac{1}{m\left(s^2 + 2\zeta\omega_n s + \omega_n^2\right)} \tag{20}$$

$$H(s) = \frac{1}{s^2 + 3.6s + 9} \tag{21}$$

In order for the rotating disk to match this reference model, the controller shown in equation 22 is proposed, where the controller parameters are to be optimized using the MIT rule [9]. The goal is to adjust these parameters so that the system mimics the reference model as closely as possible. The controller will adapt these parameters to bring the actual system closer to the reference model.

$$u = \theta_1 u_c - \theta_2 y \tag{22}$$

Also from the MIT rule, equation 23 is given, where e is the adaptation error, y is the plant output (position of the coin) and y_m is the output of the reference model

$$J(\theta) = \frac{1}{2}e^2(\theta) \quad \Rightarrow \quad \frac{d\theta}{dt} = -\gamma\frac{dJ}{d\theta} = -\gamma e\frac{\partial e}{\partial\theta} \tag{23}$$

Working with these equations, the system output is given by:

$$y = G(s)u = \frac{s^2 + 0.89s - 10.58}{-9.8}(\theta_1 u_c - \theta_2 y)$$

$$y = \frac{s^2 + 0.89s - 10.58}{-9.8\theta_1} - \frac{s^2 + 0.89s - 10.58}{9.8\theta_2} u_c$$

The error e is:

$$e = \frac{s^2 + 0.89s - 10.58}{-9.8\theta_1} - \frac{s^2 + 0.89s - 10.58}{9.8\theta_2} u_c - G_m u_c$$

From this expression, the error e can be derived, and by differentiating with respect to θ_1 and θ_2, the adaptation rules are obtained:

$$\frac{\partial e}{\partial \theta_1} = \frac{s^2 + 0.89s - 10.58}{-9.8} - \frac{s^2 + 0.89s - 10.58}{9.8\theta_2} u_c$$

$$\frac{d\theta_1}{dt} = -\gamma e \frac{\partial e}{\partial \theta_1}$$

$$\frac{\partial e}{\partial \theta_2} = -\frac{(a_1 m + a_2 m)}{s^2 + a_1 m s + a_2 m} y$$

$$\frac{d\theta_2}{dt} = -\gamma e \frac{\partial e}{\partial \theta_2}$$

Assuming that the real model with the implementation of θ_1 and θ_2 is very close to the reference model, the following approximation can be made:

$$s^2 + 0.89s - 10.58 - 9.8\theta_2 \approx s^2 + a_1 m s + a_2 m$$

From this assumption, the following derivatives are obtained:

$$\frac{\partial e}{\partial \theta_1} = \frac{a_1 m + a_2 m}{s^2 + a_1 m s + a_2 m} u_c$$

$$\frac{\partial e}{\partial \theta_2} = -\frac{(a_1 m + a_2 m)}{s^2 + a_1 m s + a_2 m} y$$

5.1 Experiments

Once this is done, the control is implemented with the diagram seen in Figure 13. With this diagram, some experiments are done. Firstly, the control of the first difficulty level goes well with no problem, which can be seen in Figure 14 (see the following link for the simulation: `https://youtu.be/Re7pplZF9PQ`). In this graphic, which will be used for the rest of the experiments, the control can be analyzed with the angle θ being externally applied to the game, the location x of the coin in the game, and the value for the end of the track in each level, which changes depending on the difficulty.

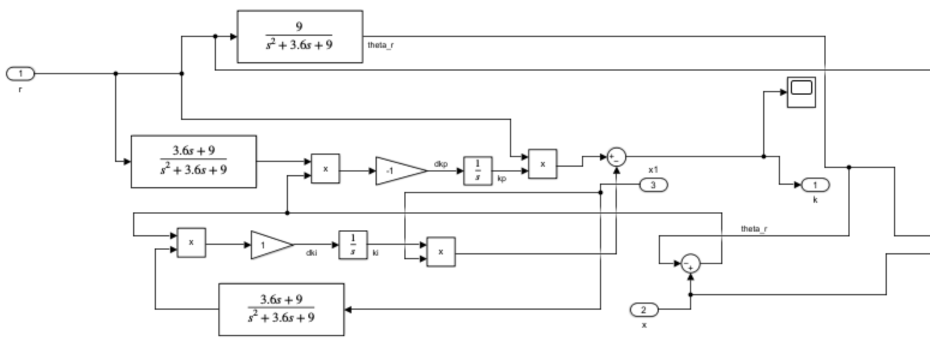


Figure 13: Simulink block diagram implemented for the Adaptive control

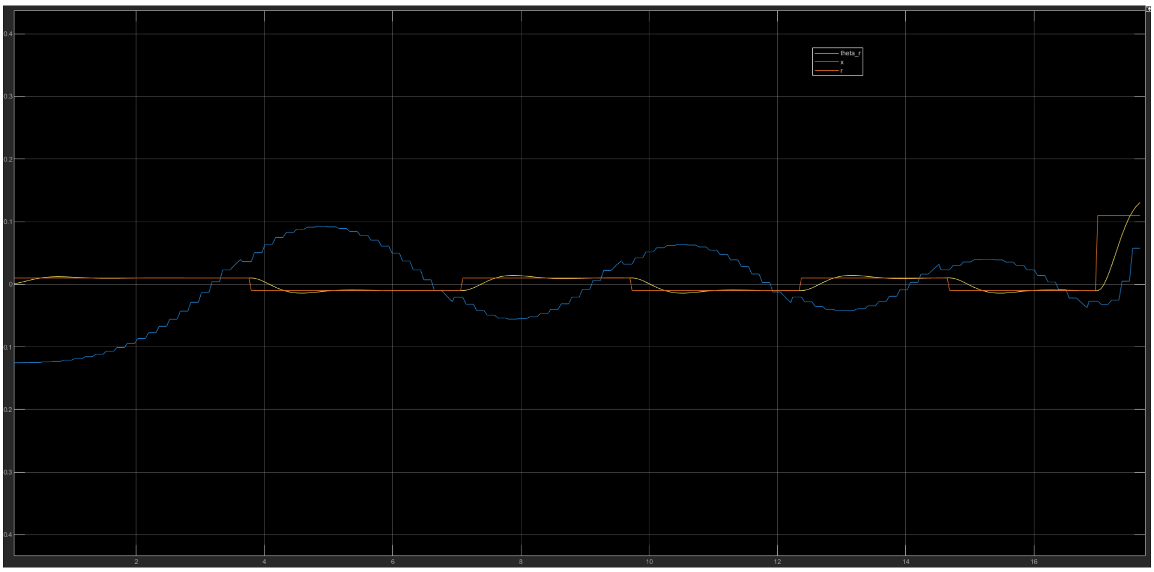
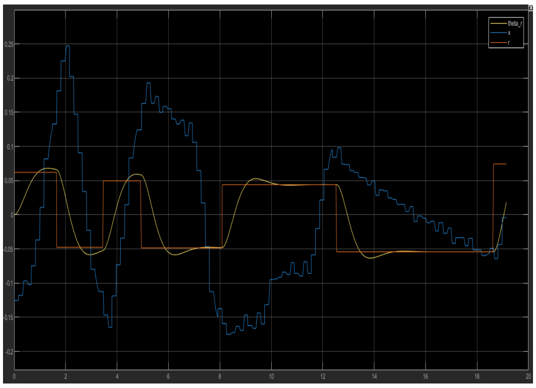
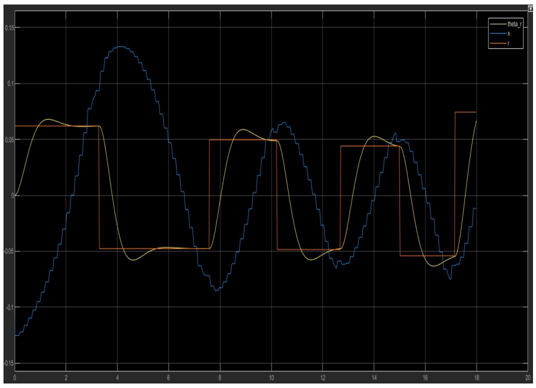


Figure 14: Graphics of the control for the first tests with difficulty 1

However, as seen in video `https://youtu.be/IMznC7B0nqg` and in Figure 15a, for the second difficulty level, the coin moves erratically and does not do a well control like previously. After some research, it is concluded that can be solved with a simple PD control [10]. With the implementation of this other controller, the coin moves in a more uniform or non-turbulent way, as can be seen in the following link: `https://youtu.be/-ibil006t-s` and in Figure 15b.



(a) Second difficulty level control with no PD controller



(b) Second difficulty level control with PD controller implemented

Figure 15: Graphics of the control for the 2nd level of difficulty

With this same PD controller, difficulty levels 3 and 4 can be solved. The parameters for the Proportional and the Derivate parts of this controller can be changed to make the coin move faster or slower, but 0.05 for D, 0.2 for P and 10 for N are good parameters to start for all levels (see equation 24 for how these parameters define a PD controller in Simulink). In Figure 16 the control of each difficulty level can be seen, with all levels been completed with the coin reaching its final destination.

$$P + D\frac{N}{1 + N\frac{1}{s}} \tag{24}$$

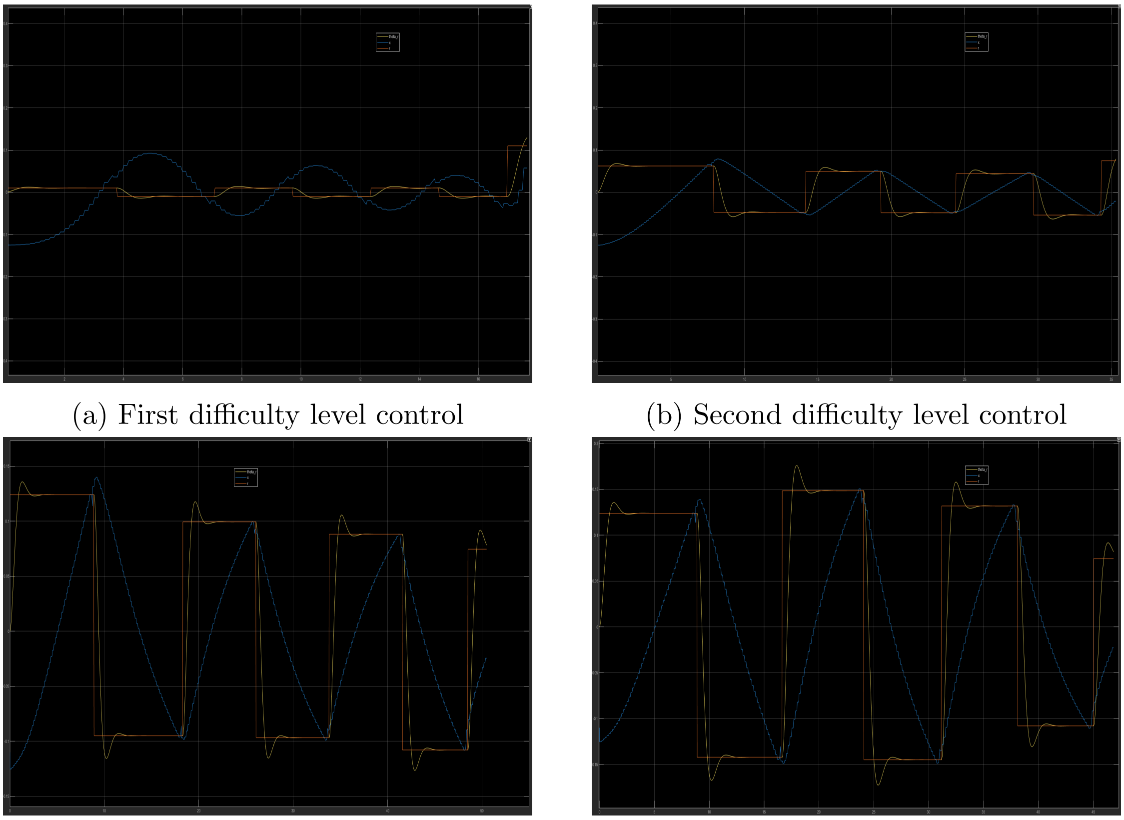
To resume, please see the different difficulties being completed in the following links:

- Difficulty 4: `https://youtu.be/xv7gOm6D_tI`
- Difficulty 3: `https://youtu.be/535esAv7464`
- Difficulty 2: `https://youtu.be/-ibilO06t-s`
- Difficulty 1: `https://youtu.be/Re7pplZF9PQ`

5.2 Conclusions and possible improvements

During this assignment, the classical Spanish arcade game Monza was controlled as part of the course on non-linear systems. This involved deducing and testing the equations governing the coin's motion. An adaptive controller was subsequently employed to manage the coin's dynamics, with simulations provided by the professor demonstrating its effectiveness.

A significant portion of the assignment was dedicated to deriving the motion equations. As the system was neither trivial nor typical, prior coursework offered limited guidance. Newton's laws and geometrical principles were applied to formulate the equations. However, adjustments were necessary to align with the professor's model, and these modifications, along with their



(a) First difficulty level control

(b) Second difficulty level control

(c) Third difficulty level control

(d) Fourth difficulty level control

Figure 16: Four different difficulty level controls completed

rationale, are detailed in this report.

The adaptive controller, as recommended by the professor, was implemented for control. This required additional learning, with references consulted and included in this work. Exploring alternative control methods, such as fuzzy controllers, neural networks, or reinforcement learning, could have been valuable. However, these topics are covered in other courses within the Master's program. This assignment, therefore, provided a focused opportunity to learn and apply adaptive control techniques in the context of a non-linear system.

Regarding the adaptive controller, it successfully managed the system at the first difficulty level. At higher difficulty levels, however, it encountered challenges, necessitating the use of a PD controller. This demonstrated how classical control methods can complement more advanced techniques in solving specific problems.

Potential improvements include developing a more robust adaptive controller capable of handling all difficulty levels without resorting to a PD controller. With additional time, further exploration of alternative control strategies could also have been pursued. Nevertheless, for the time available, the adaptive controller provided a successful and practical solution to the problem.

References

[1] H. Goldstein. *Classical Mechanics*. Pearson, 1980.

[2] Md. Arafat Hossain et al. "Control strategy review for hydrogen-renewable energy power system". In: *Journal of Energy Storage* 72 (2023), p. 108170. ISSN: 2352-152X. DOI: https://doi.org/10.1016/j.est.2023.108170. URL: https://www.sciencedirect.com/science/article/pii/S2352152X23015670.

[3] Paolo Dadone and Hugh Vanl. "Fuzzy Neural Adaptation Through Evolutionary Exploration". In: *Virginia Polytechnic Institute and State University* (1999).

[4] Mohammad Jafari and Hao Xu. "Intelligent Control for Unmanned Aerial Systems with System Uncertainties and Disturbances Using Artificial Neural Network". In: *Drones* 2 (Aug. 2018), p. 30. DOI: 10.3390/drones2030030.

[5] Karl Johan Astrom and Bjorn Wittenmark. *Adaptive Control*. USA: Addison-Wesley Longman Publishing Co., Inc., 1994. ISBN: 0201558661.

[6] Sergio Dominguez Cabrerizo. *Control Óptimo y Adaptativo*. 2020.

[7] Sergio A. Castaño Giraldo. *Sistemas de SEGUNDO ORDEN Control Explicación COMPLETA*. ouTube Video. 2020. URL: https://www.youtube.com/watch?v=YhJF1v5cEfw.

[8] MATLAB. *Model Reference Adaptive Control*. MATLAB web page. URL: https://es.mathworks.com/help/slcontrol/ug/model-reference-adaptive-control.html.

[9] Aleksandar Haber. *Introduction to Model Reference Adaptive Control with MATLAB Simulations: MIT Rule Implementation*. YouTube Video. 2021. URL: https://www.youtube.com/watch?v=5xLbtyN5XBU.

[10] Electricalvoice. *Controllers – Proportional, Integral and Derivative Controllers*. Web blog. 2020. URL: https://electricalvoice.com/controllers-proportional-integral-derivative-controllers/.