processes

MDPI

Article

Design of Aerodynamic Ball Levitation Laboratory Plant

Tomáš Tkáčik [1], Milan Tkáčik [1,*], Slávka Jadlovská [2] and Anna Jadlovská [1]

[1] Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, Slovakia; tomas.tkacik@tuke.sk (T.T.); anna.jadlovska@tuke.sk (A.J.)

[2] Department of Industrial Engineering and Informatics, Faculty of Manufacturing Technologies with a Seat in Prešov, Technical University of Košice, Bayerova 1, 080 01 Prešov, Slovakia; slavka.jadlovska@tuke.sk

* Correspondence: milan.tkacik@tuke.sk; Tel.: +421-55-602-4214

Abstract: This paper presents the development of a new Aerodynamic Ball Levitation Laboratory Plant at the Center of Modern Control Techniques and Industrial Informatics (CMCT&II). The entire design process of the plant is described, including the component selection process, the physical construction of the plant, the design of a printed circuit board (PCB) powered by a microcontroller, and the implementation of its firmware. A parametric mathematical model of the laboratory plant is created, whose parameters are then estimated using a nonlinear least-squares method based on acquired experimental data. The Kalman filter and the optimal state-space feedback control are designed based on the obtained mathematical model. The designed controller is then validated using the physical plant.

Keywords: aerodynamic levitation; experimental identification; system state estimation; linear optimal control; noise filtration; motor control

check for updates

Citation: Tkáčik, T.; Tkáčik, M.; Jadlovská, S.; Jadlovská, A. Design of Aerodynamic Ball Levitation Laboratory Plant. *Processes* **2021**, *9*, 1950. https://doi.org/10.3390/pr9111950

Academic Editor: Krzysztof Rogowski

Received: 16 September 2021
Accepted: 25 October 2021
Published: 30 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The core principle of aerodynamic levitation is based on suspending an arbitrary object in the air using airflow. The practical use-case of this phenomenon can be seen in applications where physical contact between objects must be avoided. It is usually utilized on a microscopic scale to prevent oxidation or crystal formation upon object contact with its container. This method allows studying materials under unusual conditions and even allows the formation of new glass materials [1].

The goal of our design of an Aerodynamic Ball Levitation Plant is to levitate a ping-pong ball inside a tube using the stream of air produced by a fan. From a control engineering aspect, this model is interesting for its simplicity of modeling and its low price. Similar to the Magnetic Levitation Plant, which uses a magnetic force to levitate a steel ball [2], the Aerodynamic Levitation Plant could also be a part of control engineering courses provided by the Center of Modern Control Techniques and Industrial Informatics (CMCT&II) at the Department of Cybernetics and Artificial Intelligence (DCAI), Faculty of Electrical Engineering and Informatics (FEEI), Technical University of Košice (TUKE). As the commercial availability of such plant types is limited, we opted to construct one. The main advantage of the Aerodynamic Levitation Plant over the Magnetic Levitation Plant is its slower dynamics. It allows the use of a larger sampling period and consequently more computationally expensive control algorithms. This model also provides a wider ball position operation range that makes experimental identification simpler.

Several world scientists have already worked on a similar Aerodynamic Ball Levitation Plant. The authors of [3] focused mainly on the physical construction of the plant, the choice of the most suitable sensor for measuring the ball's position, and its analytical modeling. Apart from the physical construction, the authors of [4] focused more on controller design based on the experimentally identified Autoregressive model with Exogenous Variable (ARX model) in the input–output form. In this case, an Arduino Mega equipped with an ATmega2560 microcontroller was used as an analog-to-digital interface of the control computer. Others contributed mainly to the application of a nonlinear modification of the Proportional-Integral-Derivative (PID) controller in the context of the Aerodynamic Levitation Plant [5]. Papers [6–8] were more focused on making this type of plant available in virtual and remote laboratories and used classical controllers like the PID controller. The authors of [9] had taken a more experimental approach in the controller design and utilized a simple switched controller inspired by the sliding window controller. Although the noise filtration and proper state estimation are purposefully left out in this design method, the designed control algorithm performed well in the simulation and on the real plant. Up to this point, all authors used airflow to levitate a ball inside a tube. A different approach was used in [10–12], where the ball was not placed inside a tube. This arrangement is more challenging in terms of both measuring the ball's position and the power requirements. On the other hand, the plant is inherently stable from a control engineering aspect, as the airflow speed gradually decreases over distance. The authors of [13] presented a model-free based control algorithm design for a similar plant (the Magnetic Levitation Plant). This method uses an experimental approach to fine-tune controller parameters for a target application. Another option is to use a nonlinear state-space controller as in [14]. This controller provides the robustness needed for systems with varying parameters. From our own experience [2] with a similar magnetic levitation plant, we decided to start with a linear state-space controller instead of a PID controller. Although linear controllers are not as robust as their nonlinear counterparts, they are easier to design. The authors of [15] used the exact linearization approach to design a controller for the Magnetic Levitation Plant. This method combines linear controller synthesis with nonlinear compensation. A notable drawback of this method is finding a suitable mathematical transformation that transforms the nonlinear system model into a linear one.

While facing the task of experimental identification, most authors have chosen linear approximations of a mathematical model, such as the ARX model [4,5,7,16]. From the control design aspect, this is not an issue, since they proved the ARX model to be sufficient for this application. The main limitation seen in this approach is that the nonlinear mathematical model obtained from analytical identification is more suitable for the controller verification in a simulation environment. This is the reason why the authors of [17] opted for nonlinear parameter estimation. Their focus was to create an open-source design that can be adopted by others. Although filtration is mentioned in figures, no more information about the used filtration method is provided. Considering that noise is the natural phenomenon of all physical systems, room for improvement is present. Only [16] utilized a low-pass filter to reduce sensory noise. There are a few papers [5,7,16] that included dynamics of a fan in the mathematical model, but they did not utilize state-space control. The goal of this paper is to solve presented open problems, mainly the parameter identification of the mathematical model, noise filtration, and utilization of the state-space control.

In this paper, we present the improved Aerodynamic Ball Levitation Plant with a fan speed sensor. We consider the addition of this sensor important for better plant control as it enables the use of more sophisticated control and state estimation algorithms. Furthermore, the additional sensor helps with the experimental estimation of the mathematical model's parameters. The proposed methodology of the Aerodynamic Ball Levitation Plant modeling and control design is incorporated in the entire paper layout. This includes subtasks in the following order: mathematical modeling, creation of physical plant, experiment design, data acquisition, nonlinear parameter estimation, design of Kalman filter, controller design, validation in the simulation environment, and validation on the real plant. Methodology verification is presented in this paper using sets of experiment and their results.

The paper consists of several sections. Section 2 deals with the construction of the Aerodynamic Ball Levitation Plant, whereas Section 3 describes the analytical modeling and parameter identification of the mathematical model using the nonlinear least-squares method. Section 4 deals with the design and validation of the Kalman filter and the optimal state-space controller with integrator (Linear-Quadratic-Integral control—LQI). Finally, Section 5 describes the inclusion of the laboratory plant into the Distributed Control System (DCS) architecture at the CMCT&II.

2. Hardware Design

The main reason for the construction of a custom plant over buying one was to reduce costs and to adjust all its aspects to our needs. Components like enclosure and connection fittings were designed in Fusion 360 (a 3D modeling software) and later 3D printed using a PLA filament. Our goal was to create a portable laboratory plant that can be easily moved.

Besides the already mentioned basic principles of the plant, there are a few design considerations that one needs to take into account. A sensor must be placed inside the tube to determine the ball's position. There is also a need to dynamically adjust the fan speed to achieve the desired effect of levitation. The rest of this section is designated to present the plant's physical, electrical, and firmware design.

2.1. Physical Construction of the Plant

Firstly, the type of ball for levitation had to be selected. From all available options, a ping-pong ball was chosen due to the strict standardization of its parameters. Regarding the tube selection, two constraints were considered—transparency of its walls and inner diameter to be slightly larger than the ball's diameter. A slightly larger inner diameter of the tube will allow the ball to move freely in the vertical direction while avoiding unnecessary horizontal movement. In our case, the inner diameter is about 4 mm wider than the ball. In a fan selection process, a computer fan was considered at the first, since it can be controlled using a pulse-width modulated (PWM) signal and often includes a tachometer to measure its speed. However, modern computer fans are optimized for silent operation using higher static pressure and lower air velocity. As low air velocity is undesirable for our application, we opted for a fan extracted from a hairdryer. It is powered by a 24 V DC motor and its diameter is close enough to the diameter of the tube. Additionally, it provides sufficient output air velocity to push the ball up in the tube. Because of the missing tachometer on the fan, we attached a Pololu 20D magnetic encoder to its back shaft. This modification enables the possibility to directly measure the fan's angular velocity. To power the fan, we decided to use a 19 V DC laptop charger to keep the plant portable. Finally, we added a VL53L0X laser rangefinder to measure the ball's position within the tube. This choice was motivated by the sensor's accuracy and simple interface, similar to the solution described in [3].

A simple user interface is added to the plant so that it can be used without an additional computer. This includes a 4-digit 7-segment display and a rotary encoder with a push-button that allows the user to change parameters on the fly. Additionally, a scale was engraved onto the tube with an LED backlight to easily determine the ball's position. The final 3D assembly of the plant is shown in Figure 1.

2.2. Design of Electronic Components

To make the plant operational on its own, a microcontroller was needed. For this task, the Arduino Nano development board with an ATmega328P microcontroller was chosen. Since no microcontroller can handle currents large enough to power a DC motor, additional components were necessary. As connecting all components using jumper wires is impractical, we decided to create a plant control unit—a printed circuit board (PCB) designed in the Autodesk EAGLE software. This PCB interconnects a 5 V regulator (LM7805), an NPN BJT transistor (BC547) used as a MOSFET driver, an n-channel MOSFET (IRF530) to drive the fan, and a number of JST XH connectors. Connectors are used to connect other components (motor, motor speed sensor, 7-segment display, rotary encoder, and ball position sensor) that cannot be placed onto the main PCB. To futureproof our PCB design, we added several connectors of various interfaces in case more sensors or actuators are added in the future. The Arduino Nano development board is directly soldered onto the control unit. The PCB design is shown in Figure 2.

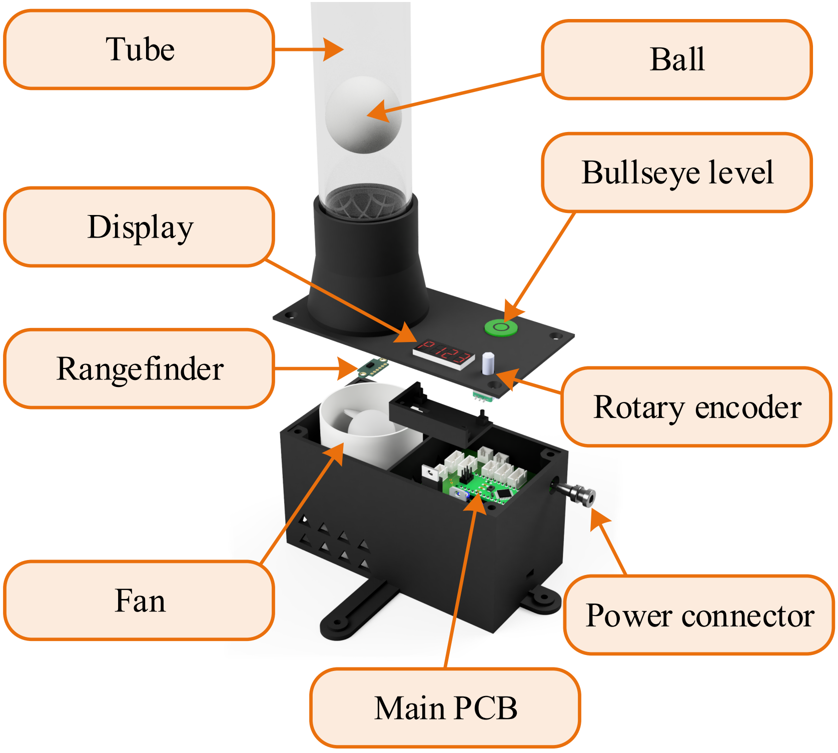

Figure 1. Component assembly of the Aerodynamic Ball Levitation Laboratory Plant.

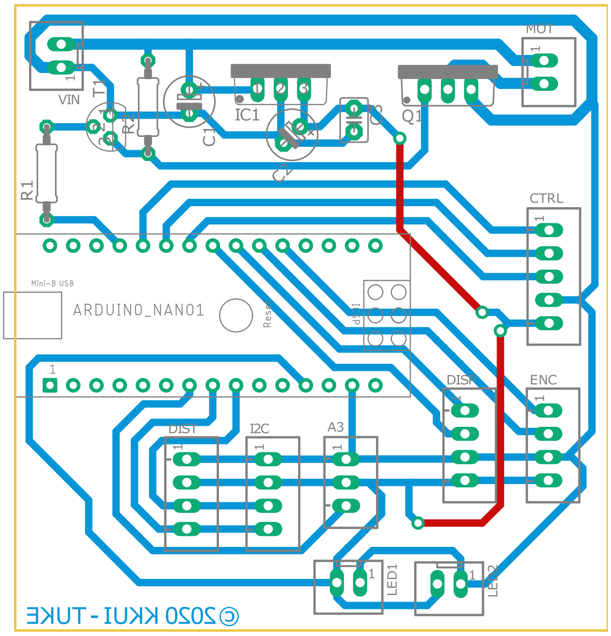

Figure 2. PCB design of plant's control unit.

2.3. Firmware Design

Our firmware design comprises four modules, as shown in Figure 3. All information exchanged between these modules is preprocessed and uses standard units where applicable. The firmware itself is implemented in the C++ language and is compiled using the Arduino IDE.

The *communication module* is used as a communication interface between a computer and the plant using a Universal Asynchronous Receiver–Transmitter (UART) serial link. For this task, a simple packet-based communication protocol was created that uses the most significant bit of each byte to control communication. In addition, a 28-bit fixed-precision data type was proposed to allow the exchange of real numbers using this protocol. The communication interface running on the computer that handles this protocol is written in the Python programming language. It is easier to work with binary data in Python than in Matlab while keeping the interface platform-independent.

The *driver module* is used as an interface between the plant and peripherals such as sensors and actuators that require the use of timers or interrupts. Its interface is programmed to work with physical units to prevent information misinterpretation. It allows for easier hardware changes as the hardware-specific code is isolated.

The *UI module* handles user input from the rotary encoder and the push-button to allow parameter changes without the need for a computer. It enables a user to enter/exit parameter editor by long button push and to change currently selected parameter using a short push. Rotation action is handled by selected parameter type: for example, a *BoolParameter* switches its state, whereas a *FloatParameter* increments or decrements its value according to rotational speed.

The *control module* estimates and filters values of the state vector using the Kalman filter. It also implements polynomial and LQI controllers, whose parameters can be changed using the mentioned communication protocol. Both implemented control algorithms inherited from the *AbstractController* interface making it easier to implement different control algorithms in the future.

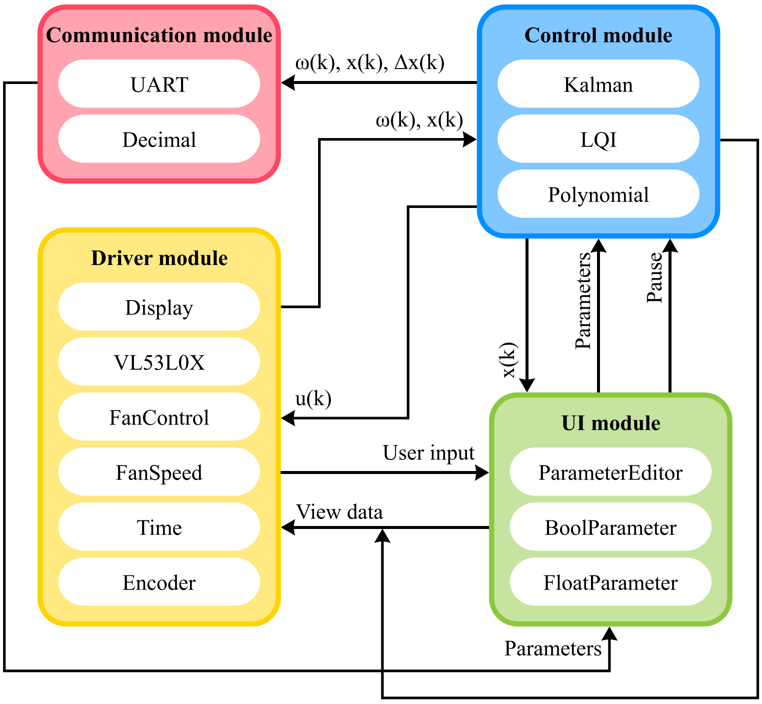

Figure 3. Firmware design diagram.

3. Mathematical Modeling

In research papers [4,9,16], a mathematical model of the Aerodynamic Ball Levitation Plant is presented as a single-input and single-output (SISO) system, with input being the fan's voltage $u(t)$ and output the ball position $x(t)$. Even though this representation does not affect the mathematical model itself, it is problematic during the experimental parameter estimation task. This is due to a high number of parameters and insufficient insight into internal system states. To overcome this limitation, our plant includes a fan speed sensor that enables us to model the system as a single-input and multiple-output (SIMO) system. To make the modeling task simpler, we are going to model two main subsystems (a fan and a ball in the tube) separately according to the diagram shown in Figure 4.

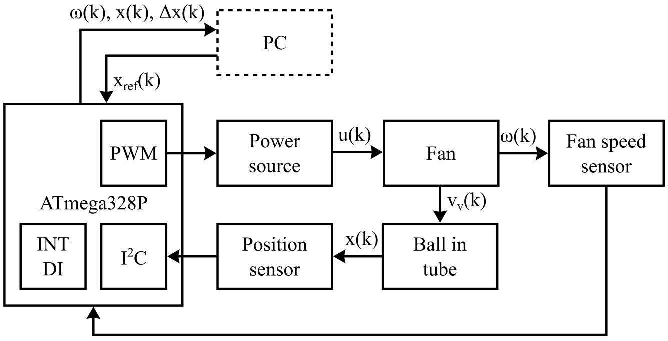

Figure 4. Diagram of subsystem interaction of Aerodynamic Ball Levitation Plant.

Since our fan is driven by a DC motor, we can create its mathematical model using the kinematic equation for the rotational motion and Kirchhoff's second law. Ignoring all nonlinearities, the resulting mathematical model is described by two first-order differential equations (Equation (1)). State variables of this model are an armature current $i(t)$ and an angular velocity $\omega(t)$ [18].

$$\begin{aligned} J\dot{\omega}(t) + b_l\omega(t) &= K_M i(t) \\ L\frac{di(t)}{dt} + Ri(t) + K_e\omega(t) &= u(t) \end{aligned} \tag{1}$$

where: J - rotor's moment of inertia [kg.m^2]
$\dot{\omega}(t)$ - rotor's angular acceleration [rad.s^{-2}]
b_l - rotor's friction constant [N.m.s]
$\omega(t)$ - rotor's angular velocity [rad.s^{-1}]
K_M - motor's torque constant [N.m.A^{-1}]
$i(t)$ - armature current [A]
L - armature inductance [H]
R - armature resistance [Ω]
K_e - motor's back electromotive force constant [V.rad^{-1}.s^{-1}]
$u(t)$ - motor's input voltage [V]

To be able to use the proposed mathematical model (1) of the fan for the parameter identification task, it is crucial to measure both of its state variables ($i(t)$ and $\omega(t)$). Having sensors to measure both state variables (armature current $i(t)$ and rotor's angular velocity $\omega(t)$), it would be possible to use state-space control algorithms to control this subsystem. However, our hardware design does not include a current sensing circuit, so a slight model modification is needed. An approximation of the fan model in form of a linear first-order differential equation (Equation (2)) was chosen for this task. Such a model has only one state variable ($\omega(t)$) that can be measured with a rotary encoder included in our hardware design. Limitations of this approximation include slightly different system dynamics, the merger of system parameters, and the use of input–output based control algorithms. The parameter merging poses a problem if the physical interpretation of parameter value is required. This is not a problem as long as we intend to create a stabilizing controller.

$$\dot{\omega}(t) + b_m\omega(t) = K_m u(t) \tag{2}$$

where: b_m - fan's damping factor [s^{-1}]
K_m - motor constant [rad.s^{-2}.V^{-1}]

The ball in the tube subsystem shown in Figure 5 can be modeled using Newton's second law if all forces acting on the ball are identified. To make it simpler, we consider only two major forces—a force of aerodynamic drag $F_o(t)$ and a gravitational force F_g that can be calculated using Formula (3).

$$F_g = m_g g \tag{3}$$

where: F_g - gravitational force [N]
m_g - ball's weight [kg]
g - gravitational acceleration [m.s^{-2}]

The aerodynamic drag force depends on a ball's drag coefficient C_g, ball's cross section area S_g, density of the air ρ_v, and square of ball's relative velocity as in (4). The relative velocity is defined as a difference between air velocity $v_v(t)$ and ball's velocity $\dot{x}(t)$.

$$F_o(t) = \frac{1}{2} C_g S_g \rho_v (v_v(t) - \dot{x}(t))^2 \tag{4}$$

where: $F_o(t)$ - aerodynamic drag force [N]
C_g - ball's drag coefficient [$-$]
S_g - ball's cross section area [m^2]
ρ_v - air density [kg.m^{-3}]
$v_v(t)$ - air velocity [m.s^{-1}]
$\dot{x}(t)$ - ball's velocity [m.s^{-1}]

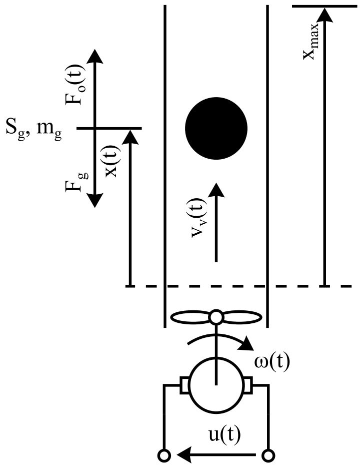

Figure 5. Simplified diagram of Aerodynamic ball levitation plant.

Since air velocity $v_v(t)$ in the tube is relatively low, all mentioned factors except the relative velocity can be combined into a single constant b_v as in (5).

$$b_v = \frac{1}{2} C_g S_g \rho_v \tag{5}$$

where: b_v - ball's drag factor [kg.m^{-1}]

According to the diagram in Figure 5, both acting forces have opposing direction of the effect (6).

$$F(t) = F_o(t) - F_g \tag{6}$$

where: $F(t)$ - force acting on the ball [N]

During normal operation, the aerodynamic drag force $F_o(t)$ is always acting in opposite direction to the gravitational force F_g. To make its direction of influence physically correct under all conditions, we added the sign function to the model Equation (7).

$$m_g \ddot{x}(t) = b_v (v_v(t) - \dot{x}(t))^2 sgn(v_v(t) - \dot{x}(t)) - m_g g \tag{7}$$

where: $\ddot{x}(t)$ - ball's linear acceleration [m.s^{-2}]
$sgn(...)$ - sign function

Finally, it is needed to find a link between the fan's angular velocity $\omega(t)$ and air velocity $v_v(t)$. According to our experiments shown in Figure 6, the air velocity $v_v(t)$ has linear dependency on the fan's angular velocity $\omega(t)$ as in (8). A constant term is added to compensate for nonlinear behavior around zero.

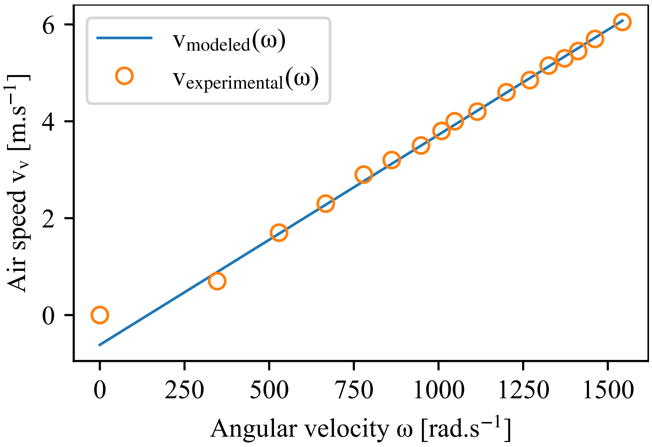

Figure 6. Air velocity $v_v(t)$ as a function of fan's angular velocity $\omega(t)$.

$$v_v(t) = K_v\omega(t) + C_v \tag{8}$$

where: K_v - fan's coefficient [m.rad^{-1}]
C_v - fan's constant [m.s^{-1}]

The complete mathematical model of the Aerodynamic Ball Levitation Plant is shown in (9). Since this model contains several unknown parameters (b_m, K_m, b_v, K_v, C_v) that cannot be measured directly, we are going to estimate their values by means of experimental identification.

$$\begin{aligned}\dot{\omega}(t) + b_m\omega(t) &= K_m u(t)\\ \ddot{x}(t) &= \frac{b_v}{m_g}(K_v\omega(t) + C_v - \dot{x}(t))^2 sgn(K_v\omega(t) + C_v - \dot{x}(t)) - g\end{aligned} \tag{9}$$

3.1. Experimental Identification and Model Validation

Experimental identification techniques can be divided by the type of resulting model into two major groups—grey-box and black-box model identification [19]. As quite a few authors [4,5,7,16] have already put their effort into black-box model estimation [20], we decided to focus on the first option and identify parameters of the plant model (9) to create a grey-box model. The general schema of the identification topology is shown in Figure 7, where in our case the approximation model is replaced with the analytical model (9). Then an optimization algorithm is used to tune the model's parameters to minimize the quadratic cost function (10). This topology can be used for both online and offline identification tasks. Due to the fast dynamics of our plant, we opted for the offline variant.

$$J(\boldsymbol{\phi}) = \frac{1}{N}\sum_{k=1}^{N}\left(\boldsymbol{e}^T(k,\boldsymbol{\phi})\boldsymbol{W}(\boldsymbol{\phi})\boldsymbol{e}(k,\boldsymbol{\phi})\right) \tag{10}$$

where: $J(\boldsymbol{\phi})$ - quadratic cost function
$\boldsymbol{\phi}$ - vector of model's parameters
$\boldsymbol{e}(k,\boldsymbol{\phi}) = \boldsymbol{y}(k) - \hat{\boldsymbol{y}}(k,\boldsymbol{\phi})$ - output prediction error vector
$\boldsymbol{W}(\boldsymbol{\phi})$ - weighting matrix

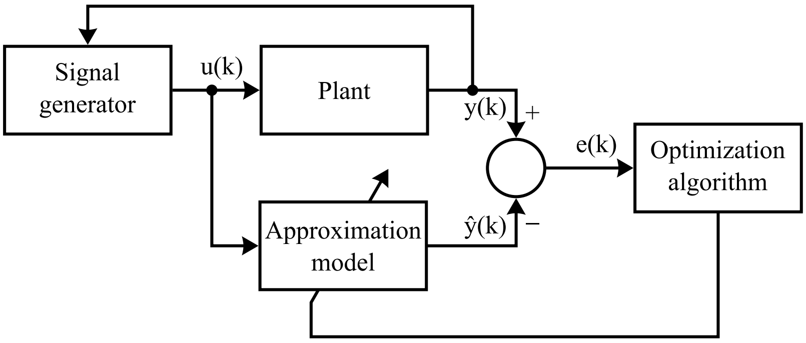

Figure 7. General schema for experimental parameter identification driven by output prediction error.

Like the modeling, the parameter identification task can be also split into two parts according to the model subsystems. From the control engineering aspect, the fan subsystem is stable, and thus the identification can be performed in the open-loop using a pseudo-random rectangular input signal with varying amplitude. Because the ball's presence affects airflow inside the tube, it was fixed inside the tube during the experiment. Obtained experimental data were used to estimate parameter values (b_m, K_m) of the fan's model using *tfest* function in MATLAB. The normalized root mean squared error of this model is 12.55% (the lower the better) and the adjusted R-squared value of this fit is 98.42%. Graphical model validation is shown in Figure 8.

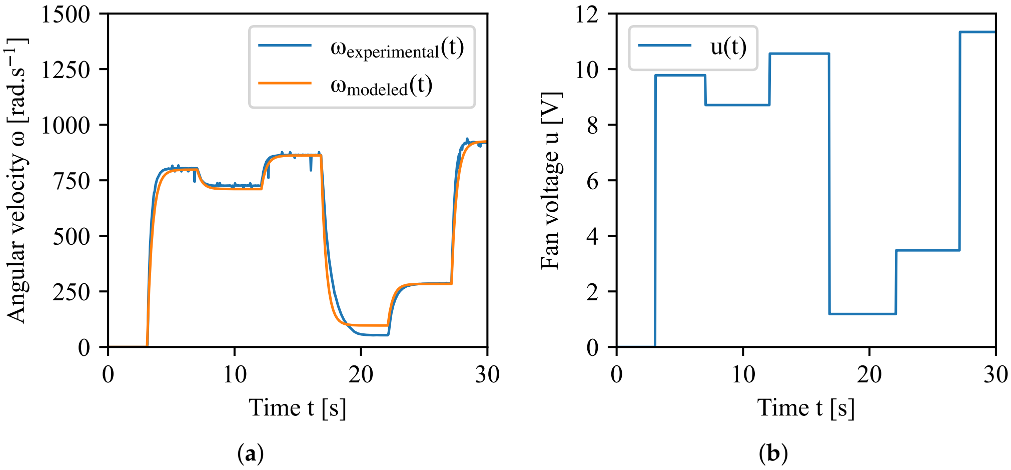

Figure 8. Fan's mathematical model validation. (**a**) Real plant compared to modeled fan angular velocity $\omega(t)$; (**b**) fan's input voltage $u(t)$.

Parameter identification of the ball in the tube subsystem is a bit more challenging because this subsystem is inherently unstable [5]. To overcome this issue, a closed-loop feedback controller must be used to avoid the ball's position saturation at its physical limits. Instead of designing a full-fledged controller, we decided to go with a simple rule-based algorithm. This algorithm oscillates the ball around the center point by switching input value between ±10% of equilibrium value. The time at which the input value is switched is always random to prevent harmonic oscillations. The parameters (b_v, K_v, C_v) of mathematical model (9) are estimated by nonlinear least-squares method using a *nlgreyest* function in MATLAB.

Results in Figure 9 do not look promising at the first glance since the ball's position of the identified model does not follow the position of the real ball precisely. This can be caused by a slight deviation in the value of some parameters and since the system behaves as an integrator, this error accumulates quickly over time. From the control engineering aspect, this should not be an issue in controller design as controllers must be robust enough to cope with system uncertainty. Values of estimated parameters are shown in Table 1.

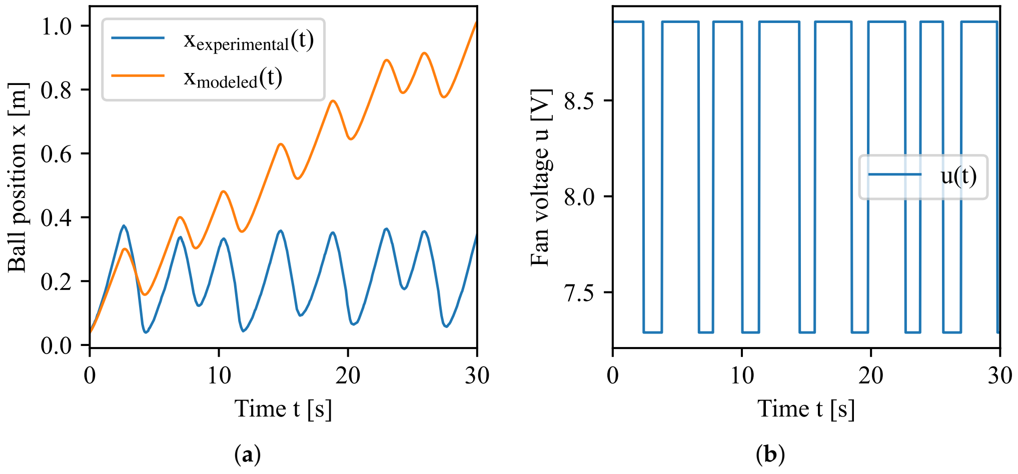

Figure 9. Validation of the ball in tube subsystem. (**a**) Real plant compared to modeled ball position $x(t)$; (**b**) fan's input voltage $u(t)$.

Table 1. Estimated model parameters.

Label	Name	Value	Unit
b_m	fan's damping factor	2.491	s^{-1}
K_m	motor constant	203.270	$rad.s^{-2}.V^{-1}$
b_v	ball's drag factor	9.202×10^{-3}	$kg.m^{-1}$
m_g	ball's weight	2.700×10^{-3}	kg
K_v	fan's coefficient	1.979×10^{-3}	$m.rad^{-1}$
C_v	fan's constant	0.369	$m.s^{-1}$

3.2. *Linearization and Discretization*

The mathematical model (9) is not suitable for designing a discrete LQI controller, because this method is based on a linear discrete model in state-space form. To get the correct form, the model (9) must be linearized first and then discretized. To make the notation of state vector cleaner, a substitution (11) is proposed.

$$\boldsymbol{x}(t) = \begin{bmatrix} \omega(t) & x(t) & \dot{x}(t) \end{bmatrix}^T = \begin{bmatrix} x_1(t) & x_2(t) & x_3(t) \end{bmatrix}^T \tag{11}$$

where: $\boldsymbol{x}(t)$ - system's state vector
$x_1(t) = \omega(t)$ - fan's angular velocity [rad.s^{-1}]
$x_2(t) = x(t)$ - ball's position [m]
$x_3(t) = \dot{x}(t)$ - ball's speed [m.s^{-1}]

The mathematical model (9) can be rewritten accordingly (12).

$$\begin{aligned} \dot{x}_1(t) &= -b_m x_1(t) + K_m u(t) \\ \dot{x}_2(t) &= x_3(t) \\ \dot{x}_3(t) &= \frac{b_v}{m_g}(K_v x_1(t) + C_v - x_3(t))^2 sgn(K_v x_1(t) + C_v - x_3(t)) - g \end{aligned} \tag{12}$$

In our case, the nonlinear model (9) is linearized at a operating point selected in the center of the tube, while other values are calculated by substituting all derivatives (9) with zero. Finally, the operating point can be defined by (13).

$$\boldsymbol{OP} = \begin{bmatrix} x_{10} & x_{20} & x_{30} & u_0 \end{bmatrix} = \begin{bmatrix} 670.819 & 0.25 & 0 & 8.222 \end{bmatrix} \tag{13}$$

where: $\boldsymbol{OP}$ - linearization operating point vector
x_{10} - operating point fan's angular velocity [rad.s^{-1}]
x_{20} - operating point ball's position [m]
x_{30} - operating point ball's linear velocity [m.s^{-1}]
u_0 - operating point fan's input voltage [V]

Linearization at the operating point $\boldsymbol{OP}$ can be performed by expanding nonlinear model (12) into first order approximation using Taylor series as shown in (14), where $\boldsymbol{y}(t)$ denotes output vector of a linear model [21].

$$\begin{aligned} \Delta\dot{\boldsymbol{x}}(t) &= \begin{bmatrix} -2.491 & 0 & 0 \\ 0 & 0 & 1 \\ 0.023 & 0 & -11.565 \end{bmatrix} \Delta\boldsymbol{x}(t) + \begin{bmatrix} 203.270 \\ 0 \\ 0 \end{bmatrix} \Delta\boldsymbol{u}(t) \\ \Delta\boldsymbol{y}(t) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \Delta\boldsymbol{x}(t) \end{aligned} \tag{14}$$

where: $\Delta\boldsymbol{x}(t) = \boldsymbol{x}(t) - \boldsymbol{x_0}$
$\Delta\boldsymbol{u}(t) = \boldsymbol{u}(t) - \boldsymbol{u_0}$

Based on the sampling period T_{sp} of the ball's position sensor ($T_{sp} = 0.04$ s), the linear model (14) is discretized using the same sampling period value. The linear discrete model in state-space form is shown in (15). This model is used to design an LQI controller.

$$\begin{aligned} \Delta\boldsymbol{x}(k+1) &= \boldsymbol{A_d}\Delta\boldsymbol{x}(k) + \boldsymbol{B_d}\Delta\boldsymbol{u}(k) \\ \Delta\boldsymbol{y}(k) &= \boldsymbol{C_d}\Delta\boldsymbol{x}(k) \end{aligned} \tag{15}$$

where: $\boldsymbol{A_d}$ - discrete linear state matrix
$\boldsymbol{B_d}$ - discrete linear input matrix
$\boldsymbol{C_d}$ - discrete linear output matrix

4. Controller Design and Verification

Thanks to the integrated fan speed sensor of our plant, a wide range of control algorithms can be utilized. Since plenty of previously mentioned authors have already worked on control algorithms based on the input-output model, the state-space control based on LQI has been selected.

Even with the addition of the fan speed sensor, it is not possible to measure the ball's speed $x_3(k)$. To overcome this limitation, some kind of estimation is needed. The simplest solution is to estimate the ball's speed $x_3(k)$ using the difference of its position $x_2(k)$. Since data from the position sensor is quite noisy, the Kalman filter is used as both an estimator and a noise filter.

4.1. Noise Filtration

The Kalman filter uses a discrete linear model (15) of the system with the addition of a system noise ($\boldsymbol{w}(k)$) and a sensory noise ($\boldsymbol{z}(k)$) (16). System noise ($\boldsymbol{w}(k)$) is used as the result of ignored or approximated parts of the physical system's dynamics, whereas sensory noise ($\boldsymbol{z}(k)$) is caused by the sensor itself or by the ADC conversion. It is expected that both system and sensory noise have a zero mean value and noise covariance matrices ($\boldsymbol{Q_E}$, $\boldsymbol{R_E}$) are known. The extended Kalman filter is a modification that allows noise filtering of non-linear systems, as system matrices ($\boldsymbol{A_d}(k)$, $\boldsymbol{B_d}(k)$, $\boldsymbol{C_d}(k)$) are being updated in every step [22]. From the perspective of the model (12), it is sufficient to use the Kalman filter (not its non-linear variant) as the change in the ball's position does not affect the linearized model, and the ball's velocity is expected to have near-zero value.

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A_d}\boldsymbol{x}(k) + \boldsymbol{B_d}\boldsymbol{u}(k) + \boldsymbol{w}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C_d}\boldsymbol{x}(k) + \boldsymbol{z}(k) \end{aligned} \tag{16}$$

where: $\boldsymbol{A_d}$ - discrete state matrix
$\boldsymbol{B_d}$ - discrete input matrix
$\boldsymbol{C_d}$ - discrete output matrix

Each iteration of the Kalman filter algorithm is divided into two phases—prediction and correction [22]. During the prediction phase, a new system state is predicted using the system's model, the previous state, and the value of the input (17). The second phase uses measurements to correct the predicted states to get an estimate of the real system state (18).

$$\begin{aligned} \hat{\boldsymbol{x}}^-(k) &= \boldsymbol{A_d}\hat{\boldsymbol{x}}^+(k-1) + \boldsymbol{B_d}\boldsymbol{u}(k-1) \\ \boldsymbol{P}^-(k) &= \boldsymbol{A_d}\boldsymbol{P}^+(k-1)\boldsymbol{A}_d^T + \boldsymbol{Q_E} \end{aligned} \tag{17}$$

where: $\hat{\boldsymbol{x}}^-(k)$ - predicted value of the state vector
$\hat{\boldsymbol{x}}^+(k)$ - corrected value of the state vector
$\boldsymbol{P}^-(k)$ - predicted value of error covariance matrix
$\boldsymbol{P}^+(k)$ - corrected value of error covariance matrix
$\boldsymbol{Q_E}$ - covariance matrix of the system noise

$$\begin{aligned} \boldsymbol{K_E}(k) &= \boldsymbol{P}^-(k)\boldsymbol{C}_d^T\left(\boldsymbol{R_E} + \boldsymbol{C_d}\boldsymbol{P}^-(k)\boldsymbol{C}_d^T\right)^{-1} \\ \hat{\boldsymbol{x}}^+(k) &= \hat{\boldsymbol{x}}^-(k) + \boldsymbol{K_E}(k)\left(\boldsymbol{y}(k) - \boldsymbol{C_d}\hat{\boldsymbol{x}}^-(k)\right) \\ \boldsymbol{P}^+(k) &= (\boldsymbol{I} - \boldsymbol{K_E}(k)\boldsymbol{C_d})\boldsymbol{P}^-(k) \end{aligned} \tag{18}$$

where: $\boldsymbol{K_E}(k)$ - Kalman gain matix
$\boldsymbol{R_E}$ - covariance matrix of the measurement (sensory) noise
$\boldsymbol{y}(k)$ - measured system output
$\boldsymbol{I}$ - identity matrix

The discrete linear model (15) was used to design a Kalman filter. Noise covariance matrices ($\boldsymbol{Q_E}$, $\boldsymbol{R_E}$) were fine-tuned using an experimental approach, so that measured noise is reduced while not affecting the dynamics of the physical plant. Exact values used are shown in (19).

$$\boldsymbol{Q_E} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}; \boldsymbol{R_E} = \begin{bmatrix} 90 & 0 \\ 0 & 30 \end{bmatrix} \tag{19}$$

In Figure 10, one can see the application of the Kalman filter in the task of noise filtration of the ball's position $x(k)$ measurement. The visible noise reduction will have a positive effect on the control of the plant.

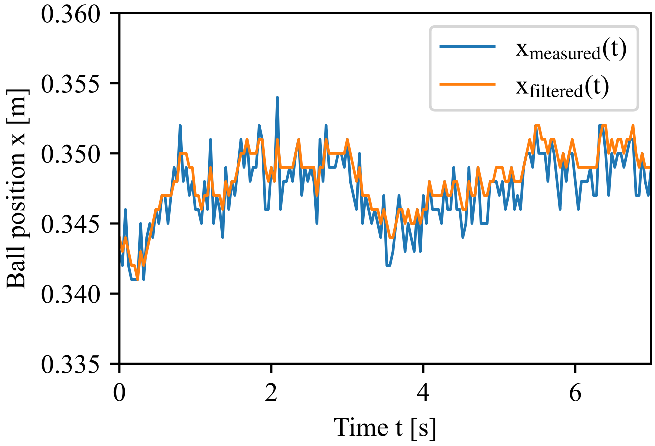

Figure 10. An application of Kalman filter in task of ball's position $x(k)$ estimation.

4.2. LQI Controller

The LQI controller is a variant of the standard Linear-Quadratic Regulator (LQR) that includes the integral term. The design of such controller starts with a discrete linear model (15) of the plant. This model is firstly converted into its extended form (20) that adds at least one state variable. How many state variables are added depends on how many state variables require zero error. Each added state variable resembles an integral of the error of the target state variable. This requires modification of the dynamics matrix $\boldsymbol{A_d}$ and input matrix $\boldsymbol{B_d}$ according to (21).

$$\Delta \boldsymbol{x_i}(k+1) = \boldsymbol{A_i} \Delta \boldsymbol{x_i}(k) + \boldsymbol{B_i} \Delta \boldsymbol{u}(k) \tag{20}$$

where: $\boldsymbol{A_i}$ - discrete linear state matrix of model in extended form
$\boldsymbol{B_i}$ - discrete linear input matrix of model in extended form

$$\boldsymbol{A_i} = \begin{bmatrix} \boldsymbol{A_d} & 0 \\ \begin{bmatrix} 0 & -T_{sp} & 0 \end{bmatrix} & 1 \end{bmatrix}; \boldsymbol{B_i} = \begin{bmatrix} \boldsymbol{B_d} \\ 0 \end{bmatrix} \tag{21}$$

where: T_{sp} - sampling period $[s]$

The LQI controller is designed in the same way as the standard LQR controller [23] by minimization of quadratic cost function (22), which is modified according to the LQI needs.

$$J(\boldsymbol{x}, \boldsymbol{u}) = \sum_{i=0}^{\infty} \left(\boldsymbol{x_i}^T(i) \boldsymbol{Q_C} \boldsymbol{x_i}(i) + \boldsymbol{u}^T(i) \boldsymbol{R_C} \boldsymbol{u}(i) \right) \tag{22}$$

where: $J(\boldsymbol{x}, \boldsymbol{u})$ - quadratic cost function
$\boldsymbol{Q_C}$ - state weighting matrix
$\boldsymbol{R_C}$ - input weighting matrix

$\boldsymbol{Q_C}$ and $\boldsymbol{R_C}$ weighting matrices are set according to (23).

$$\boldsymbol{Q_C} = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10^5 \end{bmatrix}; \boldsymbol{R_C} = \begin{bmatrix} 0.1 \end{bmatrix} \tag{23}$$

To get an optimal control algorithm based on cost function (22), the Riccati algebraic Equation (24) must be solved [24].

$$\boldsymbol{Q_C} + \boldsymbol{A_i}^T \boldsymbol{S} \boldsymbol{A_i} - \boldsymbol{S} - \boldsymbol{A_i}^T \boldsymbol{S} \boldsymbol{B_i} \left(\boldsymbol{B_i}^T \boldsymbol{S} \boldsymbol{B_i} + \boldsymbol{R_C} \right)^{-1} \boldsymbol{B_i}^T \boldsymbol{S} \boldsymbol{A_i} = 0 \tag{24}$$

where: $\boldsymbol{S}$ - solution of Riccati algebraic equation

Finally, controller (25) in form of the feedback gain $\boldsymbol{K_C}$ can be calculated using the solution $\boldsymbol{S}$ of Equation (24).

$$\Delta \boldsymbol{u}(k) = -\boldsymbol{R_C}^{-1} \boldsymbol{B_i}^T \boldsymbol{S} \Delta \boldsymbol{x_i}(k) = -\boldsymbol{K_C} \Delta \boldsymbol{x_i}(k) \tag{25}$$

where: $\boldsymbol{K_C}$ - feedback gain

Before we apply the newly designed controller (25) onto the real plant, its effectiveness needs to be verified in a simulation environment. For this task, a nonlinear mathematical model (12) implemented in MATLAB/Simulink was used. The general schema for the verification of the LQI controller is shown in Figure 11.

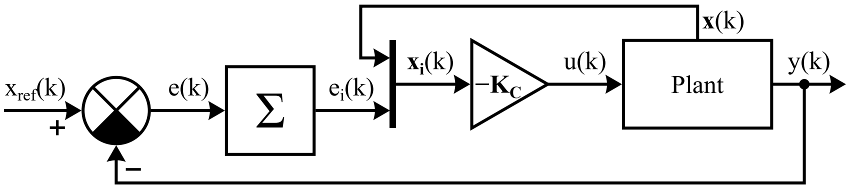

Figure 11. General schema of LQI controller.

From the simulation results shown in Figure 12, it can be concluded that the designed controller is sensitive to noise, but the regulation period is satisfactory. In this simulation, the Kalman filter was not applied while sensory noise was modeled as a standard Gaussian noise.

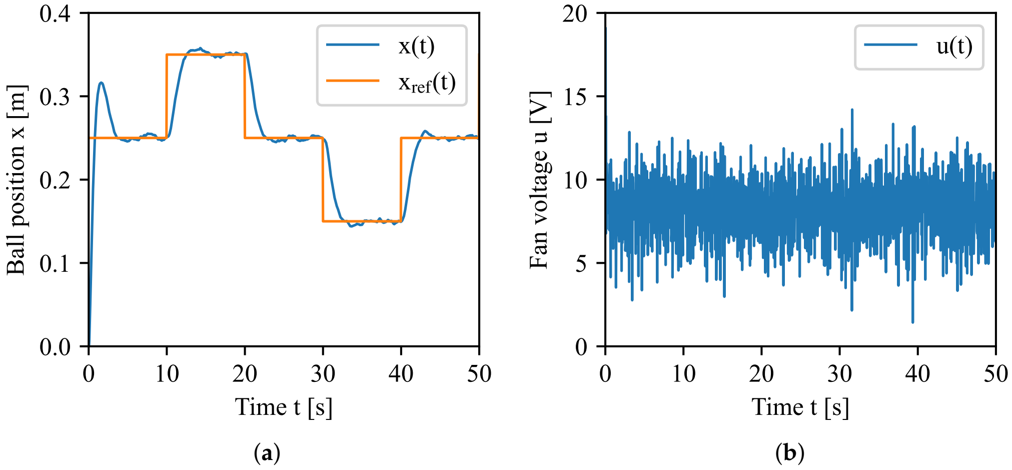

Figure 12. Simulation verification of LQI controller in Matlab/Simulink environment. (**a**) Ball position $x(t)$ tracking the reference signal $x_{ref}(t)$; (**b**) fan's input voltage $u(t)$.

After successful simulation controller verification, the controller (25) was applied on the real plant with results shown in Figure 13. The deviation of the ball's position $x(k)$ from a reference trajectory $x_{ref}(k)$ near the zero time is caused by different initial conditions between simulation and real experiment. In the case of the real laboratory plant, the initial position of the ball must be at its topmost position within the tube. This prevents the ball from being stuck near the fan due to highly turbulent airflow at its close proximity.

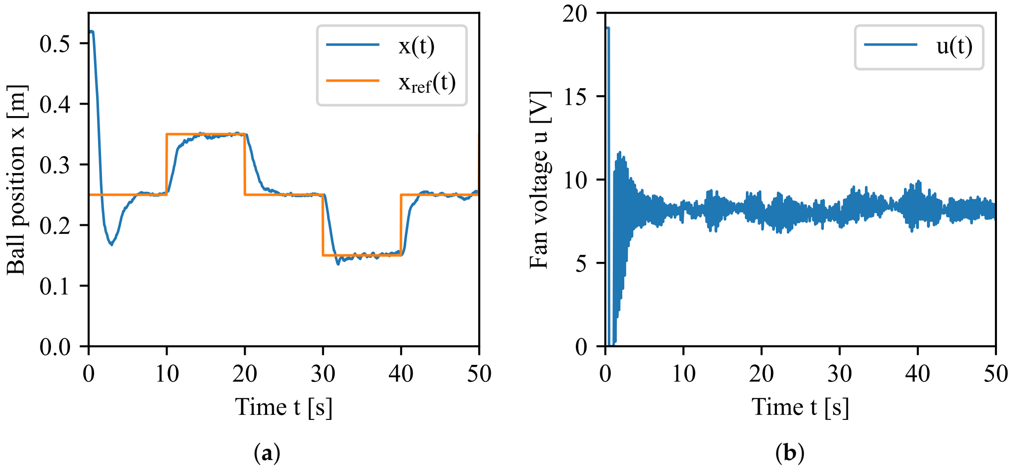

Figure 13. Verification of LQI controller on real physical plant. (**a**) Ball position $x(t)$ tracking the reference signal $x_{ref}(t)$; (**b**) fan's input voltage $u(t)$.

We also conducted position tracking experiments with a polynomial controller designed using the pole placement method and a discrete PID controller with trapezoidal integral approximation. Results shown in Figure 14 depict a comparison of these control algorithms with the previously presented LQI control algorithm. The following parameters were used in the design of these controllers:

- Polynomial controller using pole placement method (poles in continuous time): $s = \begin{bmatrix} -1 & -1.5 & -2 & -3 \pm 0.5i & -6 \end{bmatrix}$
- Discrete PID using Naslin method: $\delta_{max} = 5\% \rightarrow \alpha = 2$

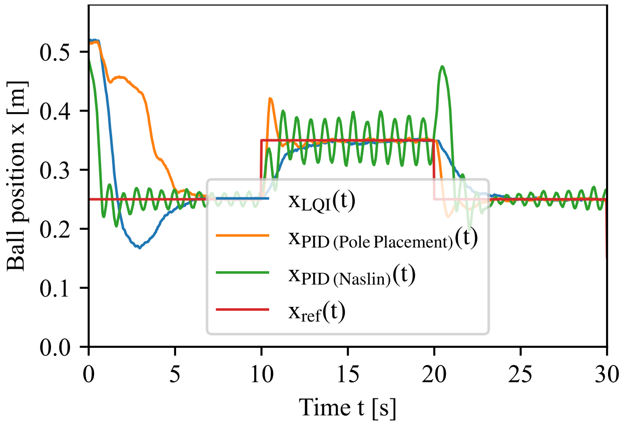

Figure 14. Comparison of various control algorithms applied on real plant.

A quantitative comparison of the selected set of control algorithms is shown in Table 2. Metric $J(\Delta e)$ quantifies how well the ball position $x(t)$ tracks the reference position $x_{ref}(t)$ and its formula is in (26). Metric $J(\Delta u)$ quantifies how stable is the fan's voltage $u(t)$ and its formula is in (27). The regulation time is the timespan it takes the ball position to reach ±5% of reference the reference signal from the start of the experiment.

Table 2. Quantitative comparison of various control algorithms on simulation and real plant.

Control Algorithm	**Regulation Time [s] (Lower Is Better)**		**Samples**		**$J(\Delta e)$ (Lower Is Better)**		**$J(\Delta u)$ (Lower Is Better)**	
	Simulation	**Real**	**Simulation**	**Real**	**Simulation**	**Real**	**Simulation**	**Real**
LQI	3.44	**5.44**	1251	1251	2.04	**2.87**	13,493.94	**1177.73**
PID (Pole placement)	3.24	5.80	1251	1251	**1.66**	4.77	**775.35**	1767.94
PID (Naslin)	**2.20**	6.12	1251	1251	1.92	5.80	56,660.30	4738.40

$$J(\Delta e) = \sum_{n=1}^{N} \Big(x(i) - x_{ref}(i) \Big)^2 \tag{26}$$

where: $x(i)$ - i-th sample of ball position
$x_{ref}(i)$ - i-th sample of ball reference position

$$J(\Delta u) = \sum_{n=2}^{N} (u(i) - u(i-1))^2 \tag{27}$$

where: $u(i)$ - i-th sample of fan voltage

5. Implementation of the Laboratory Plant within the DCS at CMCT&II

The newly created laboratory plant is implemented into a Distributed Control System (DCS) [25] at the CMCT&II. The hierarchical model of the DCS architecture shown in Figure 15 is divided into five layers of which the first three layers are currently applied to this plant. Implementation of the remaining two topmost layers of the pyramidal model is the subject of further research.

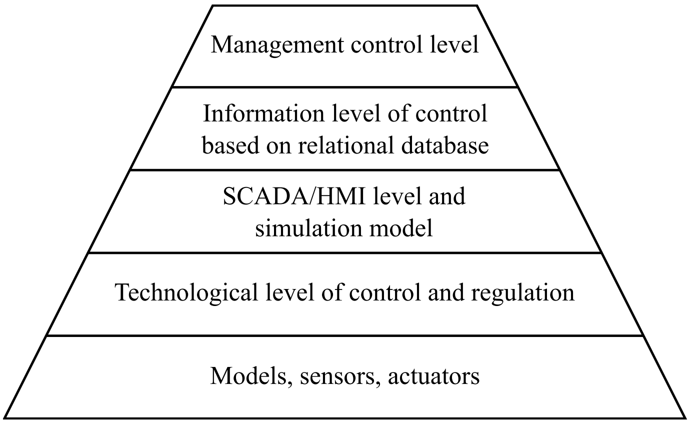
Management control level
Information level of control based on relational database
SCADA/HMI level and simulation model
Technological level of control and regulation
Models, sensors, actuators

Figure 15. Pyramidal model of the DCS architecture at CMCT&II.

The ball's position sensor, the fan speed sensor, the fan, and the whole plant's physical construction represent the zero layer of the architecture. The plant's control algorithms implemented in the ATmega328P microcontroller are part of the first layer. The second layer implements the simulation validation of control algorithms and the Human–Machine Interface (HMI) in form of the rotary encoder and the 7-segment display. The goal of this interface is to allow the user to change control parameters without the need for a computer. The schematic diagram in Figure 16 shows an implementation of mentioned layers of the DCS architecture.

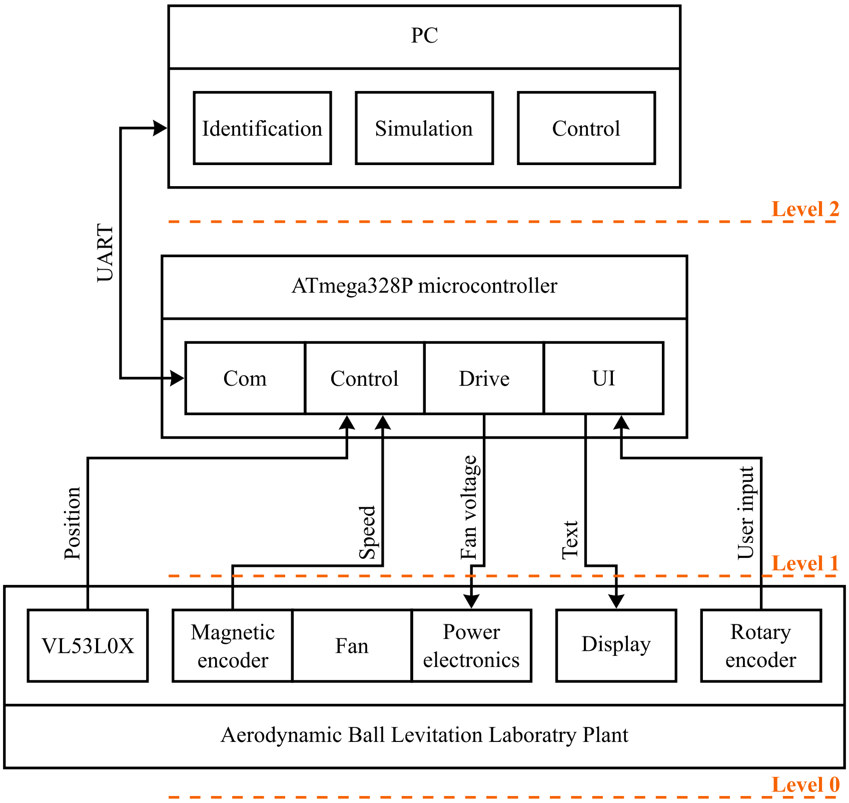

Figure 16. Implementation of the DCS architecture for the Aerodynamic Ball Levitation Laboratory Plant.

The presented hierarchical architecture is used by the CMCT&II in the ALICE (A Large Ion Collider Experiment) CERN (European Organization for Nuclear Research) research project [26,27]. The created laboratory plant is used to validate the generality of proposed solutions and further validate the concept of the DCS at the CMCT&II.

6. Conclusions

The main goal of this paper is to present obtained experimental results from the newly created Aerodynamic Ball Levitation plant. This includes the design and verification of a methodology for modeling, identification, control, simulation, and implementation on the real nonlinear laboratory plant. Additionally, details about hardware and software implementations are provided with the emphasis on design improvements. The main improvement of our design is the addition of the fan speed sensor that enabled us to use a broader range of available control algorithms. These control algorithms can be easily validated on the simulation model that acts as a digital twin for the real plant. The digital twin can be used for predictive control algorithms or to generate training data for AI-based controllers.

Another improvement was made in the use of the Kalman filter to filter the sensory noise, which in turn reduced the fluctuation of the control voltage. Lastly, the onboard control mechanism was added in form of the display and the rotary encoder with a push-button to control the plant without the need for a computer. Obtained results are presented in visual form supplemented with quantitative evaluation. The laboratory plant can be further improved, e.g., with the addition of a current sensing circuit. This circuit can improve the results of the experimental identification especially in the case of the fan subsystem. Additionally, a different set of control algorithms such as Model Predictive Control, Model Reference Adaptive Control, or Robust Tracking Control can be evaluated on this plant.

The newly created laboratory plant will be used in the education process during control engineering courses provided by CMCT&II, for example, Basics of Automatic Control, Optimal Control of Hybrid Systems, or Control and Artificial Intelligence. The plant creates a solid foundation for standard control engineering tasks, such as analytical modeling, parameter identification, and design of various types of controllers. It also enables the members of the CMCT&II to implement the laboratory plant into similar DCS architecture that is used at the CERN.

Author Contributions: Conceptualization, T.T.; methodology, T.T., M.T. and A.J. and S.J.; software, T.T. and M.T.; validation, T.T., A.J. and S.J.; formal analysis, A.J. and S.J.; investigation, T.T.; resources, T.T.; data curation, T.T. and M.T.; writing—original draft preparation, T.T.; writing—review and editing, M.T., A.J. and S.J.; visualization, T.T.; supervision, A.J. and S.J.; project administration, A.J.; funding acquisition, A.J. and S.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the project ALICE experiment at the CERN LHC: The study of strongly interacting matter under extreme conditions (ALICE KE FEI TU 0195/2021).

Acknowledgments: This publication is the result of project implementations: Modular multifunctional control workplace using computational intelligence techniques (APVV-19-0590) and ALICE experiment at the CERN LHC: The study of strongly interacting matter under extreme conditions (ALICE KE FEI TU 0195/2021).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADC	Analog to Digital Converter
ALICE	A Large Ion Collider Experiment
ARX	Autoregressive Model with Exogenous Variable
CERN	European Organization for Nuclear Research
CMCT&II	Center of Modern Control Techniques and Industrial
DCAI	Department of Cybernetics and Artificial Intelligence
DCS	Distributed Control System

FEEI	Faculty of Electrical Engineering and Informatics
HMI	Human–Machine Interface
LQI	Linear Quadratic Integral (regulator)
LQR	Linear-Quadratic Regulator
PCB	Printed Circuit Board
PID	Proportional-Integral-Derivative (regulator)
PLA	Polylactic Acid (popular material used for 3D printing)
PWM	Pulse-Width Modulated (signal)
SISO	Single-Input Single-Output (system)
SIMO	Single-Input Multiple-Output (system)
TUKE	Technical University of Košice
UART	Universal Asynchronous Receiver–Transmitter

References

1. Benmore, C.; Weber, J. Aerodynamic levitation, supercooled liquids and glass formation. *Adv. Phys. X* **2017**, *2*, 717–736. [CrossRef]
2. Šuster, P.; Jadlovská, A. Modeling and control design of magnetic levitation system. In Proceedings of the 2012 IEEE 10th International Symposium on Applied Machine Intelligence and Informatics (SAMI), Herl'any, Slovakia, 26–28 January 2012; pp. 295–299. [CrossRef]
3. Rábek, M.; Žáková, K. Building of the fan driven ball levitation system. *IFAC-PapersOnLine* **2019**, *52*, 283–287. [CrossRef]
4. Chołodowicz, E.; Orłowski, P. Low-cost air levitation laboratory stand using MATLAB/Simulink and Arduino. *Pomiary Automatyka Robotyka* **2017**, *21*, 33–39. [CrossRef]
5. Chacón, J.; Vargas, H.; Dormido, S.; Sánchez, J. Experimental study of nonlinear PID controllers in an air levitation system. *IFAC-PapersOnLine* **2018**, *51*, 304–309. [CrossRef]
6. Saenz, J.; Chacon, J.; de la Torre, L.; Dormido, S. An open software-open hardware lab of the air levitation system. *IFAC-PapersOnLine* **2017**, *50*, 9168–9173. [CrossRef]
7. Chacon, J.; Saenz, J.; Torre, L.D.l.; Diaz, J.M.; Esquembre, F. Design of a low-cost air levitation system for teaching control engineering. *Sensors* **2017**, *17*, 2321. [CrossRef] [PubMed]
8. Rábek, M.; Žáková, K. Development of Control Experiments for an Online Laboratory System. *Information* **2020**, *11*, 131. [CrossRef]
9. Chaos, D.; Chacón, J.; Aranda-Escolástico, E.; Dormido, S. Robust switched control of an air levitation system with minimum sensing. *ISA Trans.* **2020**, *96*, 327–336. [CrossRef]
10. Jernigan, S.R.; Fahmy, Y.; Buckner, G.D. Implementing a remote laboratory experience into a joint engineering degree program: Aerodynamic levitation of a beach ball. *IEEE Trans. Educ.* **2008**, *52*, 205–213. [CrossRef]
11. Kuzhandairaj, J.C. Development, control and testing of an air levitation system for educational purpose. Master's Degree Thesis, Polytechnic University of Milan, Milan, Italy, 19 April 2018.
12. Escano, J.M.; Ortega, M.G.; Rubio, F.R. Position control of a pneumatic levitation system. In Proceedings of the 2005 IEEE Conference on Emerging Technologies and Factory Automation, Catania, Italy, 19–22 September 2005; Volume 1, pp. 523–528. [CrossRef]
13. Pujol-Vázquez, G.; Vargas, A.N.; Mobayen, S.; Acho, L. Semi-Active Magnetic Levitation System for Education. *Appl. Sci.* **2021**, *11*, 5330. [CrossRef]
14. Mobayen, S.; Tchier, F. An LMI approach to adaptive robust tracker design for uncertain nonlinear systems with time-delays and input nonlinearities. *Nonlinear Dyn.* **2016**, *85*, 1965–1978. [CrossRef]
15. Barie, W.; Chiasson, J. Linear and nonlinear state-space controllers for magnetic levitation. *Int. J. Syst. Sci.* **1996**, *27*, 1153–1163. [CrossRef]
16. Tootchi, A.; Amirkhani, S.; Chaibakhsh, A. Modeling and Control of an Air Levitation Ball and Pipe Laboratory Setup. In Proceedings of the 2019 7th International Conference on Robotics and Mechatronics (ICRoM), Tehran, Iran, 20–21 November 2019; pp. 29–34.[CrossRef]
17. Takács, G.; Chmurčiak, P.; Gulan, M.; Mikuláš, E.; Kulhánek, J.; Penzinger, G.; Vdoleček, M.; Podbielančík, M.; Lučan, M.; Šálka, P.; et al. FloatShield: An Open Source Air Levitation Device for Control Engineering Education. *IFAC-PapersOnLine* **2020**, *53*, 17288–17295. [CrossRef]
18. Emhemed, A.A.; Mamat, R.B. Modelling and simulation for industrial DC motor using intelligent control. *Procedia Eng.* **2012**, *41*, 420–425. [CrossRef]
19. Brastein, O.; Perera, D.; Pfeifer, C.; Skeie, N.O. Parameter estimation for grey-box models of building thermal behaviour. *Energy Build.* **2018**, *169*, 58–68. [CrossRef]
20. Ljung, L. *System Identification: Theory for the User*, 2nd ed.; Prentice-Hall: Englewood Cliffs, NJ, USA, 1999; ISBN 978-0136566953.
21. Wang, D.; Meng, F.; Meng, S. Linearization Method of Nonlinear Magnetic Levitation System. *Math. Probl. Eng.* **2020**, *2020*, 9873651. [CrossRef]
22. Kim, Y.; Bang, H. Introduction to Kalman filter and its applications. *Introd. Implement. Kalman Filter* **2018**, *1*, 1–16. [CrossRef]

23. Young, P.C.; Willems, J. An approach to the linear multivariable servomechanism problem. *Int. J. Control* **1972**, *15*, 961–979. [CrossRef]
24. Kirk, D.E. *Optimal Control Theory: An Introduction*; Dover Publications: Mineola, New York, USA, 2012; ISBN 978-0486135076.
25. Jadlovský, J.; Jadlovská, A.; Jadlovská, S.; Čerkala, J.; Kopčík, M.; Čabala, J.; Oravec, M.; Varga, M.; Voščček, D. Research activities of the center of modern control techniques and industrial informatics. In Proceedings of the 2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI), Herľany, Slovakia, 21–23 January 2016; pp. 279–285. [CrossRef]
26. Jadlovský, J.; Jadlovská, A.; Jadlovská, S.; Oravec, M.; Voščček, D.; Kopčík, M.; Čabala, J.; Tkáčik, M.; Chochula, P.; Pinazza, O. Communication Architecture of the Detector Control System for the Inner Tracking System. In Proceedings of the 16th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS 2017), Barcelona, Spain, 8–13 October 2017; p. THPHA208. [CrossRef]
27. Tkáčik, M.; Jadlovský, J.; Jadlovská, S.; Koska, L.; Jadlovská, A.; Donadoni, M. FRED—Flexible Framework for Frontend Electronics Control in ALICE Experiment at CERN. *Processes* **2020**, *8*, 565. [CrossRef]