Research Article / Electrical-Electronic Engineering http://dx.doi.org/10.15446/ing.investig.v40n3.80124

Control of a Direct Current Motor Using Time Scaling

Control de un motor de corriente continua usando escalamiento temporal

José Danilo Rairán Antolines[1]

ABSTRACT

Humans naturally control their surrounding space. However, that capacity has not been fully used to build better intelligent controllers, mainly because the reaction time of a person limits the number of industrial applications. In this paper, the author proposes a method to overcome the problem of reaction time for a human in the control loop. This method, called Time Scaling Control, starts by modifying the constant times of the plant's model to the point where control is comfortable for a human. Then, the controller acquires the knowledge that was expressed during the human control stage and places it in a Neural Network, which controls both scaled and original plants. Time Scaling Control highly improves the control performance compared with a PID, in this case demonstrated by the control of a direct current motor, which cannot be controlled by a human without time scaling control due to the speed of the system.

Keywords: DC motor, time scaling control, plant identification, neural network, intelligent control

RESUMEN

Los humanos controlan el espacio que los rodea de manera natural. Sin embargo, esta capacidad no se ha usado completamente para construir mejores controladores inteligentes, principalmente porque el tiempo de reacción de una persona limita el número de posibles aplicaciones industriales. En este artículo se propone un método para eliminar el problema del tiempo de reacción de un humano en un lazo de control. Este método, llamado Control con Escalamiento Temporal, comienza por modificar las constantes de tiempo del modelo de la planta, hasta el punto en el que el control sea cómodo para un humano. Entonces, el controlador adquiere el conocimiento que fue expresado durante la etapa de control humano y lo ubica en una red neuronal, la cual controla tanto la planta escalizada como la planta original. El Control con Escalamiento Temporal mejora bastante el desempeño del control en comparación con un PID, demostrado en este caso por el control de un motor de corriente directa, el cual no puede ser controlado por una persona sin el uso de escalamiento temporal por la velocidad del motor.

Palabras clave: motor DC, control con escalamiento temporal, identificación de plantas, redes neuronales, control inteligente

Received: June 06th, 2019
Accepted: September 3rd, 2020

Introduction

Traditional control strategies look for a control law that properly satisfies certain conditions, such as stability, controllability and observability in a control system. These strategies can be more or less complex, according to the restrictions and requirements of the problem. However, most strategies are purely mathematical and ignore the fact that the human brain is a highly effective adaptive controller (Van der El, Pool, Van Paassen, and Mulder, 2018). Only recently have scientists begun to measure human control performance (Huang, Chen, and Li, 2015; Laurense, Pool, Damveld, Paassen, and Mulder, 2015; Inga, Flad, and Hohmann, 2017). These studies show that one of the main limitations to having a human in a control loop is time delay. Brain and body take time to provide an actuating signal and that delay could be too long for some applications.

An area of science that studies the human brain as a controller is neuroscience. Through this discipline, it is possible to explain the abilities of humans in motion planning and decision making (Mackie, Van Dam, and Fan, 2013). The majority of studies focus on the prefrontal cortex and other structures like the amygdala (Duverne and Koehlin, 2017). However, other studies show that there is a coordination among multiple brain areas, especially when the brain has last-minute decisions (Xu et al., 2017), as it happens during the control of dynamic systems. The brain is so prone to control that a new area called network control theory explains some aspects of the brain that may help in the treatment of neurological diseases (Medaglia, Pasqualetti, Hamilton, Thompson, and Bassett, 2017).

Regardless of the successes or failures of different areas of science while proving the human capacity to control, people control their environment and transform it every day. In addition, it is interesting to see that, when a human influences

[1]Electrical Engineer, Universidad Nacional de Colombia, Colombia. Ph.D. in Engineering – Systems and Computation, Universidad Nacional de Colombia, Colombia. Affiliation: Professsor at Universidad Distrital Francisco José de Caldas, Colombia. E-mail: drairan@udistrital.edu.co

How to cite: Rairán, D. (2020). Control of a Direct Current Motor Using Time Scaling. *Ingenierla e Investigación*, *40*(3). 10.15446/ing.investig.v40n3.80124

the environment it also affects the human in return. Thus, a human controlling also becomes the object of control, as can be seen in Corno, Giani, Tanelli, and Savaresi (2015). Humans have also served as inspiration for several control strategies, such as teaching robots how to control articulation to achieve natural motion (Lee, 2015), or teaching autonomous vehicles to make better decisions (Suresh and Manivannan, 2016). A broad area that has generated several proposals is human-robot collaboration, because a robot working with a human should predict human behavior, in other words, it should emulate the control actions of a human (Robla et al., 2017).

This proposal, as well as other techniques in intelligent control, aims to build better controllers by the emulation of a human during a control task. However, and first of all, the author proposes a method to eliminate the limitations caused by the time delay from the brain and body. That delay makes some systems too fast to control, whereas systems that are too slow may cause attention problems. In both cases, there may be a reduction in control performance, which limits the number of dynamic systems that a human can control. Instead, and using the proposal in this paper, a person can control any dynamic system by properly changing its speed.

The intuitive idea of the proposal, called Time Scaling Control (TSC), consists of changing the constant times of a plant until its control is comfortable enough for humans. In general, this is impossible to do with a real plant, but it is possible through the use of this model. Thus, a human controls a scaled version of the model instead of the real plant. Finally, the knowledge that was captured during the control task is learned by a neural network. An important feature of some neural network architectures, such as a Multilayer Perceptron, is that they are blind to changes in the time scale (as will be explained later), so they can be equally used for the scaled model and in the real plant.

There are three main advantages of Time Scaling Control compared with any other intelligent controller:

1. TSC captures the ability of the human brain to control dynamic systems, even when that knowledge remains hidden for the human controller, since most of the motor control activity happens unconsciously. In contrast, for instance, in Fuzzy Control, the so-called expert must be able to verbalize the knowledge to achieve control, but that is not always possible.
2. TSC allows a person to control any system, fast or slow. Current uses of the computational power of the human brain are limited to systems that evolve at a pace that matches human possibilities. Those limits disappear when using the model.
3. TSC not only allows the human to control any system, even when the person can not consciously describe the control rule, but also provides a method to transfer that knowledge into an automatic algorithm, which makes TSC ready to use in industrial applications.

The implementation of TSC requires the following three steps: *Scaling*, *Training*, *Running*:

1. *Scaling* starts by defining a model of the plant, which is then scaled, as defined in Equation 1. The application of this Equation is the only math operation during the whole control process, which is part of the beauty of this proposal.
2. During *Training*, the person controls the scaled model of the plant, and those control actions are recorded as data to train a neural network.
3. During the *Running* step, the neural network with encapsulated knowledge controls not only the scaled version of the model, but the plant itself. The three main sections of this paper (Scaling, Training, Running) detail those three steps to implement a Time Scaling Controller.

The author uses an example to show the application of the proposed control strategy: the control of the angular position of a shaft. This problem required the construction of a position sensor, as explained in the next Section. The following Section presents a central procedure in the controller design; it is the identification of the plant. The next three sections thoroughly show the step by step of the design for the proposed controller: the *Scaling* (S), the *Training* (T), and the *Running* of the controller (R). This paper ends with a discussion section, conclusions, and future work.

Angular Position Measurement

The basic configuration of a system to control the angular position of a motor requires, in addition to the controller, three components: a driver, the motor, and a sensor. The driver uses two operational LM386 amplifiers set as a bridge to allow the motor to rotate clockwise or counter clockwise. The actuating signal coming from the controller feeds the amplifier card, so that, in the steady state, the motor rotates at a speed proportional to the actuating signal.

There is a huge variety of sensors used to measure the angular position of a motor. The usual selection is an incremental encoder, but other popular choices are the absolute optical encoder and the resolver. Applications with low precision requirements, as well as small angle variations, can be measured using potentiometers. Given the high cost of an absolute encoder or a resolver, the low precision of a potentiometer, and the undesirable dependency on the initial conditions of an incremental encoder, this paper presents the design of a sensor based on the Hall effect. This sensor aims to be absolute, as well as cheap and precise.

The sensor has two components: one rotates while the other remains still. The static part has two Hall effect sensors in quadrature, plus the electronic to normalize the signal. The rotating part uses two magnets in the shaft, which produces sine and cosine type output signals in the sensors, as described in (Rapos, Mechefske, and Timusk, 2016), which is the configuration used in this paper. However, there are other interesting proposals to explore in future applications. For instance, Wu and Wang (2016) use three or six hall sensors instead of two to improve accuracy at the expense of increasing the complexity of signal processing. On the other hand, Anoop and George (2013) use two hall sensors, but propose the use of linear outputs instead of sinusoidal outputs, which may boost measurement accuracy.

The signal conditioning component shown in Figure 1 starts by filtering the noise that may appear due to mechanical vibrations or electrical interferences. The implementation of this filtering uses a traditional low-pass filter of the first order $out/in = 1/(s\tau + 1)$, which can be written using a discrete set of blocks in Simulink, following the rule $out = (1/s)(1/\tau)(in - out)$. This configuration allows the definition of an initial output for the filter as the initial condition of the Integrator $(1/s)$. That initial condition equals the initial measurement in each sensor, whereas the subtraction $in - out$ corresponds to a negative feedback connection, which passes through the constant $1/\tau$ in order to define the cut frequency of the filter (τ = 2 ms). On the other hand, the constants in the signal conditioning section in Figure 1 serve to set the average signal to zero and the amplitude to one. These values were experimentally set at c_1 = 1,094, c_2 = 1,0144, c_3 = 0,277, c_4 = 0,664. This normalization facilitates the computation of the angle made by means of a $atan2$ function. This function uses x and y to come up with and angle between $\pm\pi$, instead of the traditional $\pm\pi/2$ with the function $atan$.

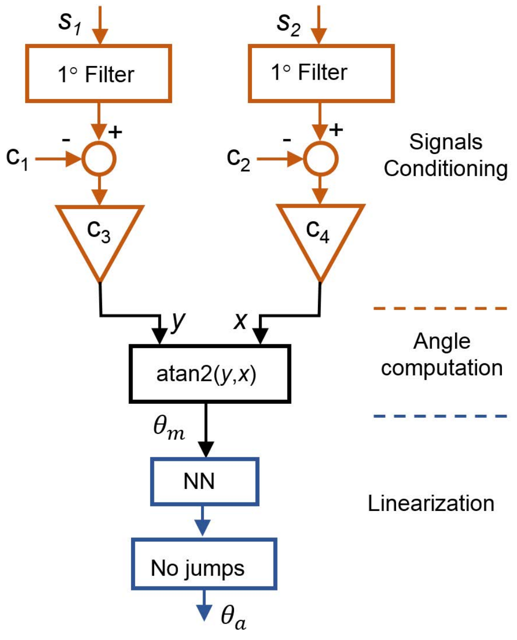

Figure 1. Block diagram of the angle estimation.
Source: Authors

Any mechanical misalignment in the measurement configuration or between the sensor gains causes a discrepancy between the actual angle, θ, and the output of the function $atan2$, also called θ_m. This difference was as high as 10%, so the author used a neural network (NN) to reduce that error. The basic idea to reduce the error is to train a NN to learn the inverse relationship between θ and θ_m. Thus, when a mechanical position θ produces an estimation equal to θ_m, that estimation ideally produces an output equal to θ in the NN. The author used a feed-forward backpropagation network with 8 neurons in the input layer and 4 neurons in the hidden layer. The input data θ cover angles between $-180°$ and $180°$, with data points every 1°. Figure 2 shows the error in the estimation using the NN to linearize the angle measurement. The error always remains under 0,3°.

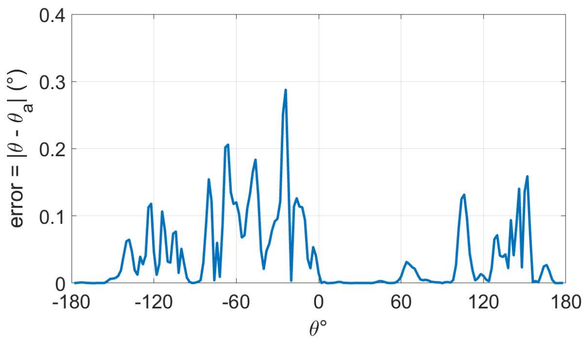

Figure 2. Error in the angle estimation.
Source: Authors

The estimation of the angle covers a single turn from -180° to 180°. Going further means generating discontinuities in the approximation due to the behavior of the function $atan2$. The final step in angle approximation implies avoiding those discontinuities, as depicted in Figure 1 by the block "No jumps". The author starts by using two new variables to detect the discontinuities: cd for jumps from 180° to $-180°$, and cu in the other case. Those constants start at zero and increase by one every time the difference between the current and the previous angle goes under $-300°$ for cd, or over 300° for cu. The value $\pm 300°$ was defined experimentally under the consideration that it should be higher than the delta angle produced at the maximum speed between two consecutive angle samples. Thus, the final approximation can be written as $\theta_a \leftarrow 360(cd - cu) + \theta_a$.

System Identification

The plant in this application is a permanent magnet direct current motor (PMDC) with a nominal current of 500 mA and nominal voltage of 12 V. This motor is widely used in robotics and it is the most popular option to test control algorithms. However, its parameters are unknown to the author. We could apply some tests to approximate the parameters of the motor, but we preferred to use the approach in this section. Firstly, because those parameters assume a linear system, when they do not; the motor can be seen as linear only around the nominal values of current, voltage, speed, and temperature. Secondly, the process used in this paper requires a single set of data to capture a model that resembles the whole dynamic instead of running several tests to obtain a model.

A traditional DC motor model uses linear differential equations to represent the relationship between the states in the motor. In this linear model, the input voltage produces an electrical current which generates a mechanical torque resulting in the rotation of the motor. Identification, in addition to the equations, requires the computation of electrical, electromechanical, and mechanical constants, such as resistance, inductance, the constant of friction, and the constant of inertia, as is presented in Wahyunggoro and Saad (2010). This type of model is used to study the motor as a dynamic system. It can also be part of a first attempt to design a controller. However the approximation given by a set of linear equations cannot fully emulate the richness of the dynamic in a real machine.

Instead of a pure linear model, Aquino and Velez (2006) propose a gray-box model. In this model, the electrical part is linear and known, whereas the mechanical part is nonlinear and unknown, as a result of the effect of changes in the load. The author proposes the use of Radial Basis Neural Networks to capture the nonlinear behavior of the model, and compare the results using a recursive two-stage method. Another approach in the identification assumes that the whole model is unknown. Thus, the linear part of the model can be represented by an ARX model, whereas the nonlinear part is chosen to be a polynomial of known order, as presented in Kara and Eker (2004). These authors study the effect of nonlinear friction and other nonlinearities in the model under the assumption that it is possible to separate the nonlinearities in two groups: static and dynamic. Another proposal by Rios and Makableh (2011) uses a Feed Forward Neural Network to capture the nonlinearities of the motor. That model is then used to simulate the effect of another neural network working as a controller of the motor. This last proposal focuses on two of the main nonlinearities studied in this paper: dead zone and saturation.

The identification of the motor in this paper uses some characteristics of the proposals in the references above, but instead of looking for a very accurate model, this paper shows that a model that approximately captures the main features of the dynamic is enough to get good control performances using time scaling. The core of the model presented in Figure 3 emulates the transient behavior of the motor using two linear blocks: one for the relation between the input voltage and the speed in the shaft (a second order transfer function, $H(s)$), and the other for the relation between speed and angular position (an Integrator, $1/s$). Given the author' experience controlling the motor, there are two main nonlinearities that should be included in its model: 1) the dead zone, and 2) the gain of the steady-state as a function of the rotation sense. The dead zone is represented by the first block in Figure 3, while the gains correspond to the triangles. The model includes another nonlinearity, the saturation of the Integrator, in other words, the natural limits for the rotation of the motor. The block between the gains and the Integrator symbolizes the selection of one gain or the other depending on the sense of rotation. If the speed is greater than or equal to zero, then the gain for the model is k_{ccw}; if the speed is negative, the gain is k_{cw}.

This is the list of parameters for the model, according to the model in Figure 3:

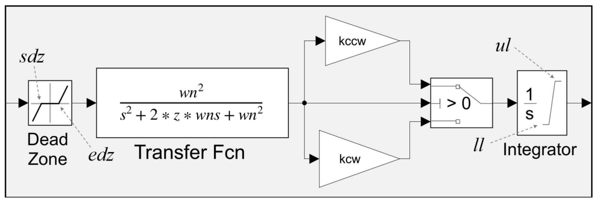

Figure 3. Motor model scheme.
Source: Authors

- Inside the block "Dead Zone", the start of dead zone, sdz, and the end of the dead zone, edz.
- Transient, in the block "Transfer Fcn": natural frequency, ω_n, damping ratio, $z = \zeta$.
- Gains: counter clockwise sense, k_{ccw}, clockwise sense, k_{cw}.
- Saturation, inside the block "Integrator", the upper limit, ul, and the lower limit, ll.

The computation of these parameters starts by defining the last two: ul and ll. Even when the control rank equals ±90°, the limits are greater than that rank. A first extra 10° allows the human controller to make mistakes at the ends of the rank and still generate good data to train the controller. A second extra 10° bounds the parameter estimation algorithm to vary the parameters of the model. Thus, the control rank is ±90°, but the human control covers ±100°, while the estimation algorithm has a window of ±110°. Thus, $ul = 110°$ and $ll = -110°$. The next two parameters correspond to the transfer function $H(s)$ in the block "Transfer Fcn". These two parameters define the shape of the transient when the input of the model changes. A way to evaluate these values implies generating sudden changes in the input of the motor and then measuring the speed of the shaft. The values that match those transients better are rounded to $\omega_n = 100$ rad/s, and $\zeta = 1$.

The four remaining parameters of the model were computed using the tool Parameter Estimation of Simulink. Given that the main goal of the model is to emulate the system when the controller is working, then a proportional controller with $k_p = 1/10$ leads the motor during the acquisition of the input and output data. The whole experiment lasts a minute and uses samples every millisecond, so there are 60 001 data points for the input and the same number for the output. The reference for the control system is a step function randomly changing from 0,3 to 0,7 s with also random amplitude from −100° to 100°. This signal was smoothed using a first order filter with constant time of 0,1 s, which better emulates the work of the motor during normal conditions. The optimization method is the Nonlinear Least Squares and the algorithm is Levenberg-Marquardt. The tolerance in the optimization is 0,001. A final estimation was run, including all the parameters except the saturation constants. The natural frequency changes to $\omega_n = 300$ rad/s. It is important to report that the change from 100 to 300 in ω_n didn't alter the quality of the approximation in more than 0,1%, which means that the nonlinearities affect the dynamic of the motor more than the linear part. The results of the optimization are shown in Figure 4, where the lower part corresponds to the input voltage, while the upper part is the output of the plant and the model.

A traditional modelling process for a DC motor defines the model as a linear transfer function, and that approximation may be useful when working around the nominal conditions of the machine. However, the DC motor for this application changes the nominal speed depending on the rotation sense, as expressed by means of the gains k_{cw} and k_{ccw}. These gains can be evaluated as the quotient between the speed in °/s at nominal voltage in V, for both positive and negative cases of voltage, but, in this case, Simulink made the approximation based on the data in Figure 4. The result is that $k_{cw} = 940°/s/V$ whereas $k_{ccw} = 530°/s/V$. Thus, the motor is almost two times faster in a clockwise sense than in the counter clockwise sense.

The last two constants of the model express the fact that the voltage in the motor requires to pass a certain threshold before the shaft starts to rotate, mainly given the friction effect. An initial estimation of those constants can be made by increasing the voltage to the point that the shaft starts to rotate, but, once more, that approximation was made by Simulink based on the data in Figure 4. The rotation sense changes the threshold, so the model requires the estimation of that value in both senses, as defined by using the gains sdz and edz inside the block "Dead Zone". The values for those gains are $sdz = -0{,}9$ V and $edz = 0{,}6$ V.

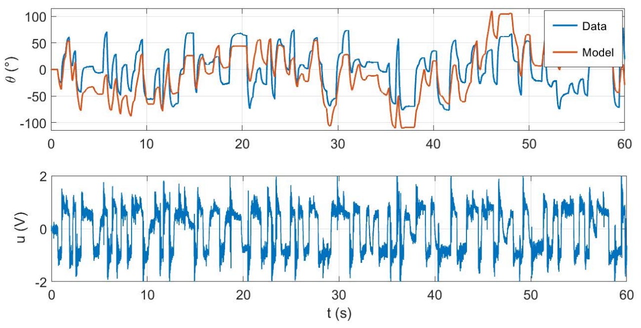

Figure 4. Output data and simulated output under the same input.
Source: Authors

In summary, the model in Figure 3 has dead zone $sdz = -0{,}9$, $edz = 0{,}6$; transient $\omega_n = 300$, $\zeta = 1$; gains $k_{cw} = 940$, $k_{ccw} = 530$; and saturation: $ul = 110°$, $ll = -110°$.

Scaling Stage (S)

The model in Figure 3 uses the traditional definition of time, but that dynamic is too fast for a human to control. Thus, this section starts by scaling the linear components of the motor, as defined in Equation (1), according to the presentation in Rairán (2017). It can be seen that the nonlinear components are not time-dependent, but constant, so the nonlinear part remains equal regardless of the time scale. The constant k_t in Equation (1) corresponds to the scaling factor. For instance, $k_t = 5$ means that the new dynamic H' is five times faster than the dynamic H. The new plant, H', keeps amplitude and shape of H and only changes how long a transient lasts.

$$H = \frac{a_n S^n + a_{n-1} S^{n-1} + \cdots + a_1 S + a_0}{b_n S^n + b_{n-1} S + \cdots + b_1 S + b_0}$$

$$H' = \frac{a_n S^n + a_{n-1} S^{n-1} k_t + \cdots + a_1 S k_t^{n-1} + a_0 k_t^n}{b_n S^n + b_{n-1} S k_t + \cdots + b_1 S k_t^{n-1} + b_0 k_t^n} \tag{1}$$

Defining the scaling factor implies an experimental procedure where the human controlling the system changes that factor until achieving a value that makes control of the system comfortable. In this case, this value is set to $k_t = 1/30$, which means that the scaled system is 30 times slower than the original. In this way, for instance, a transient of two seconds in the motor lasts a whole minute in the scaled model. The transient $H = 300^2/(s^2 + 600s + 300^2))$ corresponds to $H' = 100/(s^2 + 20s + 100)$, and the integral $\theta(s)/Vel(s) = 1/s$ corresponds to $\theta'(s)/Vel'(s) = (1/30)/s$. Both input and output in each transfer function need to be scaled as indicated by the notation. The slow human reactions produce a slow velocity, which in turn produces a slow change in the angular position. The scaling additionally requires the scaling of the sampling time at which the data is recorded or generated, from t_s to $t'_s = t_s/k_t$ in this case, $t_s = 1$ ms, so $t'_s = 30$ ms.

This stage in the control process requires interaction between the human and the scaled system. The human senses and evaluates the behavior of the plant, and, with experience in the manipulation of the system, compares his or her perceptions with the set point. As a result, the human makes two decisions: 1) whether to increase, decrease or maintain the actuating signal, and 2) how significant this change will be. The application of the actuating signal as input for the scaled system produces changes in the output of the scaled plant in such a way that, ideally, the perceived output matches the set point in the shortest time possible. The instantiation of the control stage in this paper uses a computer and a human, as shown in Figure 5.

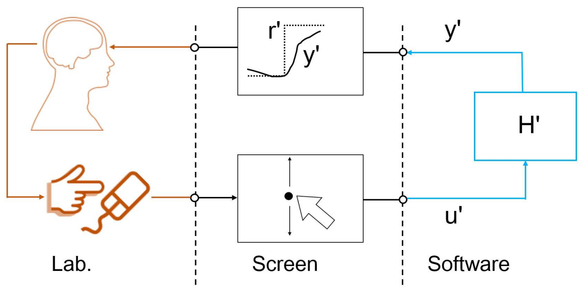

Figure 5. Control Scheme.
Source: Authors

This stage aims to generate the best data to train a NN in the next stage. Thus, the definition of the reference signal should make the system as dynamic as possible. In this case, the author defines a square signal oscillating between 100° and −100°, lasting between 12 and 15 seconds in each value. That signal passes through a low pass filter of the first order with $\tau = 2$ s, as shown in Figure 6. This filter produces references along the rank of ±100°, instead of the extreme values alone. The constant time τ aims to emulate the normal work of the motor during a traditional control application, and at the same time its stabilization time falls just below the transition time, which allows the system to reach its steady state. The data acquisition lasts 2,5 minutes. Thus, every reference value appears about five times, which is enough data to train a NN.

Figure 6 also shows that the information presented to the human controller through the scope is not the scaled error, e' (as it is done in traditional control), but its negative, $-e'$. It can be seen that, regardless of the value of the output y', the output will eventually match the sign of the input u'. For instance, if u' increases, then y' (as well as $-e' = y' - r'$) will eventually also increase. Thus, if $-e'$ is, for instance, negative, a good control action will consist of setting u' as a positive value. This action is intuitive for a human: it is performed to counteract the signal in the screen in order to have null error. On the other hand, the block "Mouse Pointer Reading" in Figure 6 runs a function based on the function "gpos" of Matlab, which provides the coordinates of the pointer of the mouse. The y value of the pointer corresponds to the value of u'.

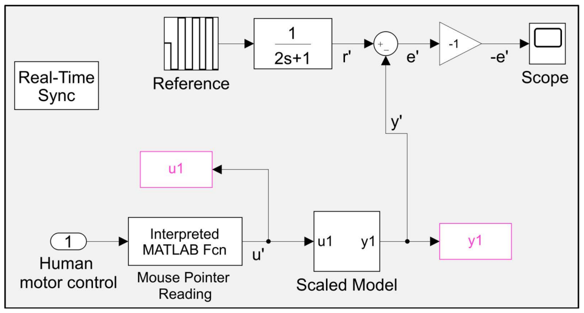

Figure 6. Control system in Simulink.
Source: Authors

Blocks $u1$ and $y1$ in Figure 6 save the data generated during the running of the simulation and make it available for use in the workspace of Matlab. The signal u' corresponds to the actions of the human controller, while y' provides the corresponding emulated position given by the scaled model. Finally, the block "Real-Time Sync" allows the running of the simulation in real time using the Simulink Desktop Real-Time toolbox.

Training Stage (T)

This stage uses the data coming from the previous stage to train a NN. Thus, instead of the human controlling the motor, the system uses the NN to define the actuating signal. The training stage starts by normalizing the data, which implies cutting and scaling. By cutting, the author understands triming and deleting the first 15 seconds of the experiment, given that this is the time required for a person to accommodate, concentrate, and be focused on the control problem. The scaling consists in dividing the input e by 180, as shown in Figure 7. This scaling makes the rank of the input [−1.1, 1.1], which is close to the rank of the activation function of each neuron [−1, 1]. Nevertheless, the effectiveness of the training remains equal under the scaling or without it in this application. The scaling makes a difference when looking at multiple inputs, as could be the case of future work, when the problem requires (for instance, in addition to e', its derivative). Normalization makes all the inputs have a similar influence during the training.

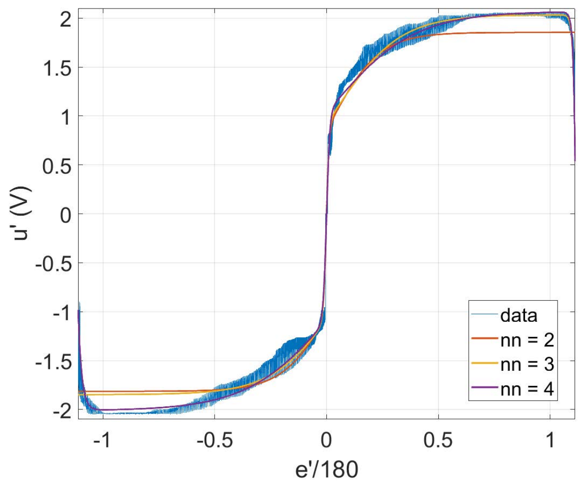

Figure 7. Effect of the number of neurons in the training.
Source: Authors

An analysis of the sorted data in Figure 7 shows the relationship between the error and the actuating signal. That relation looks like a tangent sigmoid function. This observation favors the use of a Multilayer Perceptron Neural Network to learn the given data, so that each of its neurons may have a tangent sigmoid transfer function. It can be seen that sorting error evinces the control strategy of the human, which was hidden inside the brain of the human controller until now. It is possible to divide that strategy in three zones: 1) small errors, from 0 to 0,2; 2) medium errors, from 0,2 to 0,8; 3) major errors, from 0,8 to 1,1. Zone 1 presents the highest rate of change and happens where the majority of control takes place. Zone 2 has a smaller slope and resembles the saturation of a tangent sigmoid. Zone 3, contrary to the intuition, decreases the actuating signal. This last behavior is caused by the reaction time of a human due to large errors (positive or negative). Unlike a pure mathematical algorithm, the control interface (including the brain itself) does not allow the human to change a decision instantly. However, a positive aspect of that delay is that it reduces the overshoot in the response.

The author used the Levenberg-Marquardt algorithm to train the NN, with a maximum of 1 000 epochs, and data randomly divided into three sets: 70% to train, 15% to validate the training, and 15% to test the NN. Figure 7 shows the result of changing one of the most influential parameters during the learning process: the number of neurons in the network.

Results in Figure 7 regard networks with a single hidden layer with two, three, or four neurons. Two neurons miss the third zone. Three neurons resulted better than two, but the generalization for major and negative errors fails. The best result in Figure 7 uses four neurons. That network properly matches and generalizes the data across the three zones.

Figure 8 shows the results when training 100 neural networks with four neurons in the hidden layer. The lower part of the figure shows the final mean square error mse for each training. This error was used as the measure of quality during the learning process. It can be seen that the variation of mse is not big: the best mse is 0,015, while the worst is 0,022. The upper part of Figure 8 presents the results for the NN with the best performance out of the 100 trials. This part of the figure shows the output data $u1$ and the output of the network ($u1$NN) when fed with the same input. The network closely follows human control behavior, especially in Zones 2 and 3.

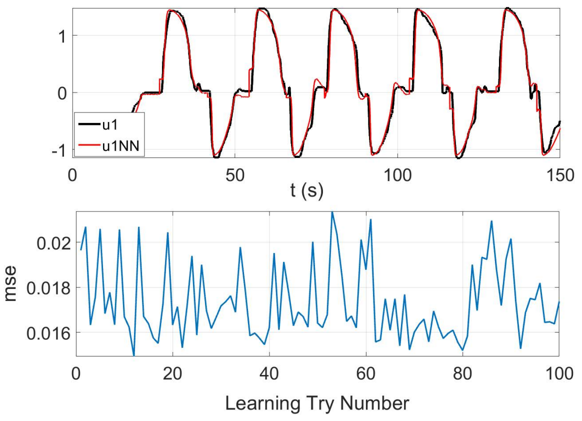

Figure 8. Selection of the best neural network.
Source: Authors

Running Stage (R)

This final time scaling control stage uses the best Neural Network of the previous stage. The best network defines an actuating signal based on the error value in order to control the models and the motor itself. The same network can control models and motor since the Multilayer Perceptron Neural Network is blind to changes in the scale of time. This type of NN produces the same outputs if the same inputs are given, regardless of time variation. If an input changes, the corresponding output is updated after a few mathematical operations. These operations are instantaneous in practice, given the speed of the systems in contrast with the time it takes to propagate a network. Thus, the same network controls the simulated system and the real plant.

Testing the trained NN starts with running the control over the scaled model H' because the controller was trained using data from H'. The test during the scaling stage lasts 2,5 minutes, which is enough time to train the network. However, the simulation in this stage may last longer, that is, 30 minutes or more. It can be seen that there is no risk that the network will stop paying attention, as it actually happens with humans after a few minutes. In addition, instead of two unique values $\pm 100°$, it is better to test several and random amplitudes in the rank $\pm 90°$. If the control works properly, the next step in testing consists of using the same network over system H. This last simulation could take a single minute, because the system runs at its normal speed (remember that a minute at the unscaled time corresponds to 30 minutes using the scaled time, $kt = 1/30$). Thus, the transitions between one amplitude and another may last from 0,3 s to 0,6 s. If this test succeeds, it is possible to run the final test, which is to control the real plant. If not, the *Training* stage should be run again.

The final and definite test uses the trained NN to control the real plant. The block diagram of the connections inside the computer is shown in Figure 9. It can be seen that the input of the network is scaled dividing the error by 180. The output of the controller reaches the plant using a data acquisition card, the National Instruments PCI-6024E. This card translates the actuating signal inside the computer into voltage to feed the power amplifier that drives the motor. Figure 9 also shows a traditional PID controller with the same tuning that was used during the *Scaling* stage, it is a proportional controller with $k_p = 1/10$.

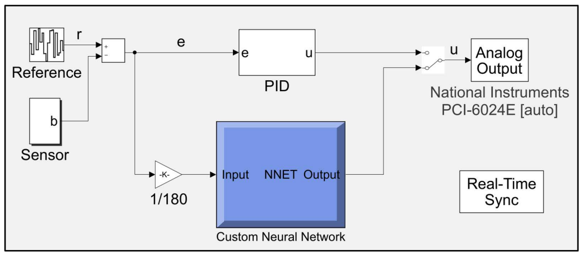

Figure 9. Block diagram to control the real plant.
Source: Authors

Figure 10 shows the performance of the Time Scaling Controller during the control of the real plant. The upper part of the figure presents the reference, the output with a PID controller, and the output with the proposed controller in this paper. TSC presents a lower overshoot and reaches the reference better, as can also be seen in the middle figure. It has a null error for the steady state after almost all the transients, but the PID has a steady state error of about 4°, this is due to the effect of the dead zone. The lower part of Figure 10 shows the actuating signal for both controllers. It is important to remark that TSC does not reach the limits of ± 2, but the PID controller does it. This characteristic reveals that the trained NN uses the energy to control the plant better than the PID for the experiment in the figure. The switching of the actuating signal for TSC happens because the controller reaches null error, so that the signal jumps alternatively from negative (or positive) to a null value. The actuating signal for the PID does not switch because it almost never reaches the null error. Finally, a measure of the performance for both controllers is the index Integral of the Absolute Error (IAE). For a trial of ten seconds, the result for the PID is 56,8 and 37,8 for TSC. Thus, the ratio between the two controllers is about 1,5 in favor of the Time Scaling Controller.

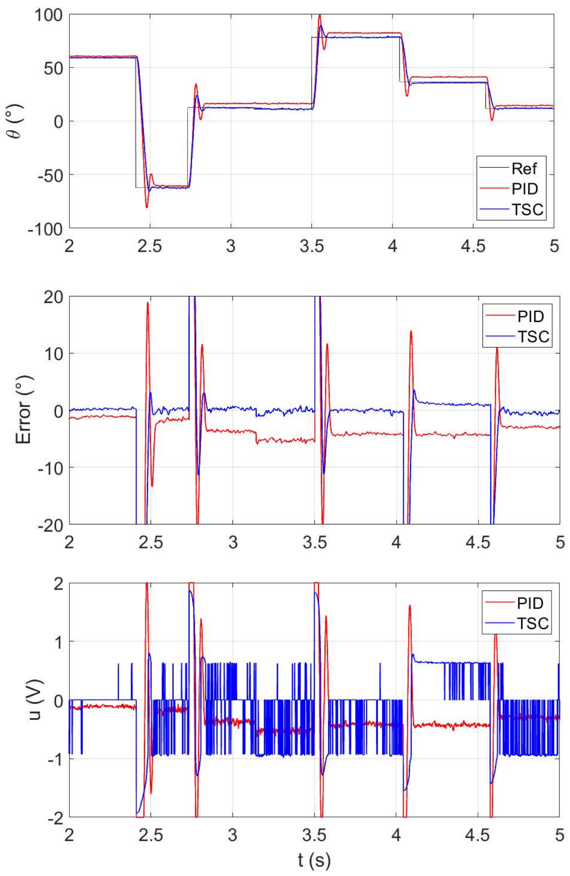

Figure 10. Control performance comparison.
Source: Authors

Discussion

Results in Figure 10 show the advantage of using TSC to regulate the position of a motor in contrast with a PID controller. However, a tuning process could improve the PID control performance, so that the apparent advantage of TSC could decrease. On the other hand, it is also true that the model of the plant could be improved, as well as the training of the NN. Thus, the ratio of 1,5 when comparing controller performances in the previous section is not definitive. Instead of a numerical ratio, this paper aims to show how to use the scaling time as a key concept in controlling dynamic systems, more specifically a motor. Both PID and TSC can be optimized, but that is not the purpose of this paper.

An aspect that can be studied to improve the performance of TSC is the influence of the human controller in the generation of data to train a NN. The data used in this paper comes from the author as the controller of the scaled model, but another person could generate different control actions, thus resulting in a different NN. Thus, TSC performance can be improved by looking at data coming from different human controllers. Another aspect that affects TSC performance is the model used to get the data. The model in Figure 3 was enough to train a NN to control the real plant, as can be seen in Figure 10. However, the difference between the output of H', H, and the real plant shows that it is possible to consider another model. It was evident that the real plant presented overshoot, while the models do not. In addition, the real plant was faster than its simulated counterpart.

The oscillation of the control signal $u(t)$ in the lower part of Figure 10 shows another feature of the proposed controller that was not programmed intentionally but emerged from the definition of the system itself. If the absolute value of the error is smaller than 10°, the actuating signal $u(t)$ switches between two values, which resembles the work of a PWM when switched to increase the efficiency of a system. TSC applies energy to the system only when it is necessary and the error is larger than 1° in this application. By doing so, the controller avoids system oscillation. However, if the absolute value grows larger than 10°, the actuating signal has a traditional continuous shape. These two control actions will be the focus of future studies on the application of TSC.

Finally, an important aspect of TSC is how the trained NN controls systems at different time scales. You can see that the network itself does not have any reference to time. However, given that the network in this paper is implemented in a computer, the network updates its output at the sample rate t_s. In this paper, $t_s = 1$ ms, whereas $t'_s = 30$ ms. Thus, and given that the stabilization time of the motor is about 20 ms, or 600 ms for H', then the updating of the network can be considered to happen instantly. Consequently, the actuating signal changes at the same speed of the error, and the error changes at the same speed of the feedback signal coming from the sensor ($e = y - r$). Thus, the actuating signal changes at the rate of the plant's output y. In summary, the controlled plant itself defines the speed of the whole system, which is not defined by the NN. If the plant is fast, the actuating signal changes accordingly; on the contrary, if the plant is slow, the network also produces a slow actuating signal.

Conclusions

The experiments in this paper show that Time Scaling Control properly controls the angular position of a motor and even results in better control performance than a PID controller. The main reason for this is that TSC uses the control knowledge of a human, which daily handles nonlinear situations and is adaptive and robust in comparison with a fixed and linear rule such as the PID controller.

The human plays the main role in TSC because he directly influences the outcome of three processes: first, defining the model of the system, as shown in Figure 3 and Figure 4; second, setting the experiment to acquire data, as shown in Figure 5 and Figure 6; and third, being the source of the knowledge to train a NN, as shown in Figure 7 and Figure 8. In the case of stages 1 and 2, the knowledge is explicit: the human is an expert in the system. In 3, TSC extracts hidden knowledge inside the brain, which may be the main contribution of TSC.

There are still many interesting topics to explore in TSC, but the author will provide just three of them as an example:

1. The definition of the minimal components of a model in order to have enough information to run TSC.
2. The definition of a deterministic method to select the best scaling factor k_t, according to the system model.
3. The analysis of data coming from different persons and the study of the effect of those differences in TSC performance.

References

Anoop, C. S. and George, B. (2013). New Signal Conditioning Circuit for MR Angle Transducers with Full-Circle Range. *IEEE Transactions on Instrumentation and Measurement, 62*(5), 1308-1317. 10.1109/TIM.2012.2236778

Aquino, A. and Velez, M. (2006). *Gray-Box Modeling of Electric Drives using Recursive Identification and Radial Basis Functions.* Paper presented at IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics, Paris, 1447-1452. 10.1109/IECON.2006.348111

Corno, M., Giani, P., Tanelli, M., and Savaresi, S.M. (2015). Human-in-the-Loop Bicycle Control via Active Heart Rate Regulation. I*EEE Transactions on Control Systems Technology, 23*(3), 1029-1040. 10.1109/TCST.2014.2360912

Duverne, S. and Koehlin, E. (2017). Rewards and Cognitive Control in the Human Prefrontal Cortex. *Cerebral Cortex, 27*(1), 5024-5039. 10.1093/cercor/bhx210

Huang, J., Chen, Y., and Li, Z. (2015). *Human operator modeling based on fractional order calculus in the manual control system with second-order controlled element*. Paper presented at The 27th Chinese Control and Decision Conference (2015 CCDC), Qingdao, 4902-4906, 10.1109/CCDC.2015.7162802

Inga, J., Köpf, F., Flad, M., and Hohmann, S. (2017). *Individual human behavior identification using an inverse reinforcement learning method*. Paper presented at 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, 99-104. 10.1109/SMC.2017.8122585

Kara, T. and Eker, I. (2004). Nonlinear modeling and identification of a DC motor for bidirectional operation with real time experiments. *Energy Conversion and Management, 45*(1), 1087-1106. 10.1016/j.enconman.2003.08.005

Laurense, V. A., Pool, D. M., Damveld, H. J., Paassen, M. R. M., and Mulder, M. (2015). Effects of Controlled Element Dynamics on Human Feedforward Behavior in Ramp-Tracking Tasks. *IEEE Transactions on Cybernetics, 45*(2), 253-265. 10.1109/TCYB.2014.2324037

Lee, D. (2015). *Incremental robot skill learning by human motion retargetting and physical human guidance*. Paper presented at the 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Goyang, 585-586. 10.1109/URAI.2015.7358837

Mackie, M. A., Van Dam, N. T., and Fan, J. (2013). Cognitive Control and Attentional Functions. *Brain Cognitions, 82*(3), 301-312. 10.1016/j.bandc.2013.05.004

Medaglia, J., Pasqualetti, F., Hamilton, R., Thompson, S., and Bassett, D. (2017). Brain and Cognitive Reserve: Translation via Network Control Theory. *Neuroscience and Biobehavioral Reviews, 75*(1), 53-64, https://10.1016/j.neubiorev.2017.01.016

Rairán, J. D. (2017). *Control of Dynamic Systems Using Time Scaling*. *ASME 2017 Dynamic Systems and Control Conference, 1*(1), 1-10. 10.1115/DSCC2017-5048

Rapos, D., Mechefske, C., and Timusk, M. (2016). *Dynamic sensor calibration: A comparative study of a Hall Effect sensor and an incremental encoder for measuring shaft rotational position*. Paper presented at 2016 IEEE International Conference on Prognostics and Health Management (ICPHM), Ottawa, ON, 1-5. 10.1109/ICPHM.2016.7542858

Rios, F. and Makableh, Y. F. (2011). *Efficient position control of DC Servomotor using backpropagation Neural Network*. Paper presented at 2011 Seventh International Conference on Natural Computation, Shanghai, 653-657. 10.1109/ICNC.2011.6022230

Robla, S., Becerra, V. M., Llata, J. R., González, E., Torre, C., and Pérez, J. (2017). Working Together: A Review on Safe Human-Robot Collaboration in Industrial Environments. *IEEE Access, 5*(1), 26754-26773. 10.1109/ACCESS.2017.2773127

Suresh, P. and Manivannan, P. V. (2016). *Human driver emulation and cognitive decision making for autonomous cars*. Paper presented at the International Conference on Robotics: Current Trends and Future Challenges (RCTFC), Thanjavur, 1-6. 10.1109/RCTFC.2016.7893411

Van der El, K., Pool, D. M., Van Paassen, M. R. M., and Mulder, M. (2018). Effects of Preview on Human Control Behavior in Tracking Tasks with Various Controlled Elements. *IEEE Transactions on Cybernetics, 48*(4), 1242-1252. 10.1109/TCYB.2017.2686335

Wu, S. T. and Wang, Z. L. (2016). Equilateral Measurement of Rotational Positions with Magnetic Encoders. *IEEE Transactions on Instrumentation and Measurement, 65*(10), 2360-2368. 10.1109/TIM.2016.2578579

Wahyunggoro, O. and Saad, N. (2010). *Analysis and evaluation of real-time and s-domain model of A DC servomotor*. Paper preseted at 2010 International Conference on Intelligent and Advanced Systems, Kuala Lumpur, Malaysia, 1-5. 10.1109/ICIAS.2010.5716206

Xu, K. Z., Anderson, B. A., Emeric, E. E., Sali, A. W., Stuphorn, V., Yantis, S., and Courtney, S. M. (2017). Neural Basis of Cognitive Control over Movement Inhibition: Human fMRI and Primate Electrophysiology Evidence. *Neuron, 96*(6), 1447-1458. 10.1016/j.neuron.2017.11.010