A Neural Network Clamping Force Model for Bolt Tightening of Wind Turbine Hubs

E.L. Secco, C. Deters, H.A. Wurdemann, H.K. Lam, *Senior Member, IEEE*, L.D. Seneviratne, *Member, IEEE*, and K. Althoefer, *Member, IEEE*

***Abstract* — Industrial manufacturing of large-scale wind turbines requires the accurate tightening of multiple bolts and nuts, which connect the ball bearings - supporting wind turbine blades - with the hub, a huge mechanical component supporting blades pitch motion. An accurate tightening of bolts and nuts requires uniformly distributed clamping forces along flanges and surfaces of contact between hub and bearings. Due to the role of friction forces and the dynamics of the phenomenon, this process is nonlinear and currently performed manually; it is also time consuming, requiring high-cost equipment and expert operators.**

This paper proposes a set of neural networks, which infer the clamping force achievable with a tightening tool while fastening M24 nuts on bolts. The tool embeds a torque sensor and shaft encoder, therefore two types of inputs of the neural networks are considered in order to fit the clamping force output: the time signals of (a) the applied torque of the tool and (b) the combination of the torque and of the angular speed of the tool.

According to results, neural networks properly model the clamping force, both during the training stage and when exposed to unseen testing data. This approach could be generalized to other industrial processes and specifically to those requiring repetitive tightening tasks and involving highly nonlinear aspects, such as friction forces.

***Index Terms* — self-adaptive manufacturing, bolt tightening, wind turbine, neural network.**

I. Introduction

NOWADAYS, the green-energy market and, in particular, that of wind turbines is expanding because of the growing, public sensitivity towards the responsible generation and usage of energy, the increased attentiveness of policy makers towards producing energy whilst avoiding pollution as well as the increasing performance and operative life of modern wind turbines [1-4]. Today's wind turbines - capable of generating power in the order of 5-6 MW - consist of large-scale wind blades and mechanical components: the rotor can be as long as 150 m and the mass of its central part, the hub, can be in excess of 20,000 kg [5]. This set-up complicates the installation process and also leads to difficulties when inspecting or repairing the plant. Energy providers aim at keeping repairs to a minimum and reduce errors that may be made during assembly; assembly errors are widely considered as contributing strongly to mechanical faults in installed wind turbines [6-8]. Usually the assembly is performed by operators, often with many years of experience: however, the overall assembly of a wind turbine may not be performed satisfying criteria concerning accurate clamping force and appropriate distribution of this force across large components such as the hub easily requiring the individual tightening of more than 100 nuts on bolts. Automating this process by means of robotics would be desirable. A few examples following such a robot-based approach are described in the literature [9-13].

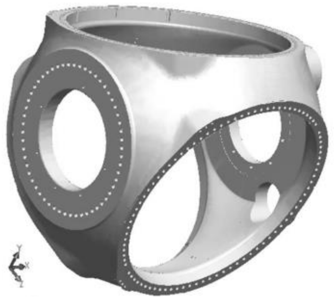

Fig. 1. Design of a wind turbine hub requiring tightening of bolts and nuts for the assembly of ball bearing which support the blades pitch motion.

One key component of the rotor is the wind turbine hub (Fig. 1): its role is to support the blades and a pitching mechanism which allows blades rotating around their longitudinal axes to adaptively follow the wind speed change while preserving an optimal spinning of the rotor [14]. This task is accomplished with hydraulic actuators and ball bearings mounted on the hub flanges to support the blades. Ball bearings are coupled with the flanges through multiple bolts and nuts (in some cases, in excess of 100 for each bearing, i.e. for each blade).

Because of the dynamic loads occurring over the operative life span of the turbine, a appropriate tightening process of these bolts and nuts during the assembly is essential as well as

Manuscript received September 15th, 2014. The research leading to these results has received funding from the European Commission's Seventh Framework Program (FP7-NMP-2009-SMALL-3, NMP- 2009-3.2-2) project COSMOS (Grant No: 246371-2.C).

E.L. Secco (corresponding author; phone: +44(0)151.291.3113; e-mail: seccoe@hope.ac.uk) is with the Department of Mathematics & Computer Science, Liverpool Hope University, UK. C. Deters is with ….. H.A. Würdemann, H.K. Lam, L.D. Seneviratne, and K. Althoefer are with the Centre of Robotics Research, Department of Informatics, King's College London, UK.

the task needs to be properly optimized. The main expected outcome of this process is to achieve a uniformly distributed clamping force along the connecting faces of the major wind turbine parts, such as the flanges of the hub.

The process is intrinsically highly non-linear, because of the high contribution of friction forces during the tightening, caused by the interactions between the threads of the bolts and nuts during the spin and, later on, by the friction occurring between the bolt and the flange during the final tightening stage [15, 16]. Other nonlinearities are introduced from the mechanical uncertainties and from the variability of the metal properties of the threads as well as from changes of temperature and humidity occurring in the working and operative environments [17-22].

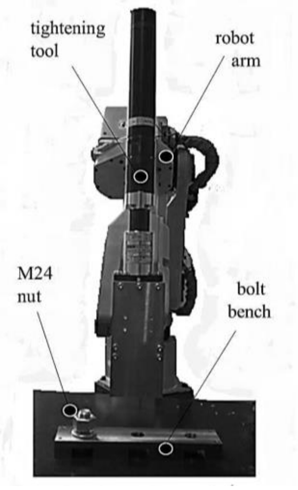

Fig. 2. Industrial robot (Fanuc M6iB) equipped with the tightening tool (BL 57/140 MDW).

A further critical and intriguing aspect of the this particular tightening process is the fact that - although the main task of the tightening is the achievement of a desired final clamping force - usually no direct measurement of this force distribution is possible in situ, because of the mechanical and geometrical configuration of the plant. One indirect approach is based on measuring the shortened length of the bolt by using, for example, ultrasonic techniques [23, 24]. Nevertheless, this approach increments the process cost requiring working time and additional equipment.

Various models of bolt tightening have been already reported in the literature: some of them were model-based or model-free controllers [25-27], the latter bypassing the need to estimate the parameters of the physical model [28-31]. For instance, in [25], authors presented the equations of screw insertion torque in function of the screw itself, the hole and the properties of the material; then, a theoretical model was validated by comparing experimental data with predictions of the model, providing basis for computerized monitoring of screw fastenings. Other approaches attempt to tailor the values of the model parameters according to physical observations: Izumi et al. [32] developed a finite element analysis approach describing the interactions between the threads of bolts and nuts during their tightening and showing as previous theory overestimates the tightening torque. Fuzzy-based tightening controllers were also proposed [15], which are capable of supervising the whole tightening process while detecting errors including cross-threading, screw jamming and nut misalignment [28-29, 33].

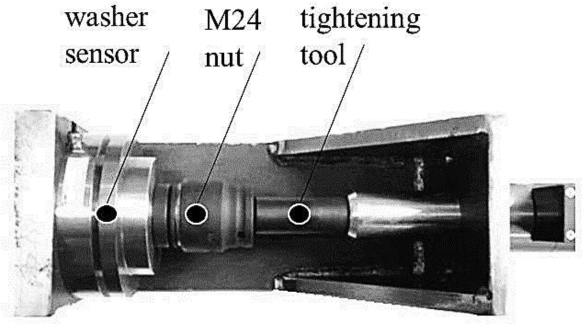

Fig. 3. Details of the experimental set-up: tightening tool (DSM BL 57/140 MDW) embeds sensors to monitor applied torque (τ) and angular displacement (α), while fastening M24 bolts and nuts (Fig. 5, top panels); a washer sensor (MecSense KMR 50 kN), captures the effective clamping force (F_{clamp}) applied to the flange (Fig. 5, bottom panel).

Performing automatic tightening with an accurate model of the clamping force - consistent with the mechanical properties of the materials and with the kinematics and dynamics of the process - remains one of the main goals of bolt tightening – and, because of the stringent build requirements for wind turbines, this is specifically the case for the manufacturing of wind turbines.

Recognizing the advantages of automated methods involving numerically controlled screwdrivers and nut tighteners, this paper proposes an experimental set-up and a neural network based approach ensuring the correct clamping force between nut and flange after the tightening is complete, improving on current strategies, which rely on humans to execute the tightening procedure by employing hand-held wrenches and tensioning tools. Apart from improving the quality and repeatability of the tightening process, the proposed approach is also likely to decrease the cost of the manufacturing as well as its execution time [6, 7, 34, 35].

The paper is organized as follows: Section II reports on the materials and methods, experimental design and neural network based approach, Section III details the results, finally Sections IV and V report the discussion and conclusion, respectively.

II. Materials & Methods

A. Experimental design

An instrumented tightening tool, model BL 57/140 MDW (by DSM Messtechnik GmbH) is used for the tightening of M24 bolts and nuts. Due to the need of an automatized process, the tool is mounted on an M6iB Fanuc Industrial Robot (Figs. 2-3). The set-up also includes a three M24 bolt bench to allow the bolt tightening of a restricted numbers of bolts and generalize the process on the wind turbine hub. Data are acquired via a set of instruments and sensors as it follows.

The tightening tool is equipped with an analogue encoder with an accuracy of 1° - for the real-time monitoring of tool and nut angular displacement during the tightening – and a torque sensor with an accuracy of 1% of the end value – for the real-time acquisition of the torque, as it is applied by the tool to the fastening nut. During the tightening process, nuts are run down a bolt and tightened against a metal flange (Fig. 3): in order to monitor the resultant clamping force, a washer sensor, model KMR 50 kN (MecSense Kraftmesstechnik), with an accuracy of 0.5%, is positioned between the nut and the flange. This sensor is usually not employed in an industrial set-up and allows a better accuracy of the measurement compared to ultrasound instrumentation (see Section I).

TABLE I
DATA DISTRIBUTION (10 TRIALS)

	t	α	τ	ω	F_{clamp}
	[s]	[deg]	[N·m]	[deg/s]	[kN]
mean	0.52	1974.50	162.04	4743.60	33.13
2·std	0.04	52.13	17.51	220.27	5.30
%	6.70	2.60	10.80	4.60	16.00

The tightening process involves forces of friction between the threads of the bolt and the nut as well as between the bolt and the flange (Section I): according to this characteristics, a theoretical model of bolt tightening has been previously reported in [16, 36], showing the relationship between the torque and the clamping force in function of the bolt and nut geometry and two coefficients of friction, namely the contact between the male and female threads (μ_t) and the friction between the nut and the flange (μ_b). In general, it is not easy to estimate these coefficients straight away and a mechanical characterizations - based on the nominal properties of the materials which are involved in the tightening - do not return a realistic estimation of the physical parameters: according to material properties compendium [37], for untreated steel, the coefficient of friction for screw threads (μ_t) is estimated between 0.12 and 0.18; similarly, for untreated steel, the coefficient of friction between the nut-bolt face against the clamped surface (μ_b) is in a range between 0.10 and 0.18. These forks of values do not allow a proper and consistent assessment of the real tightening dynamics, furthermore because of the presence of another mechanical element, which is the aforementioned washer sensor.

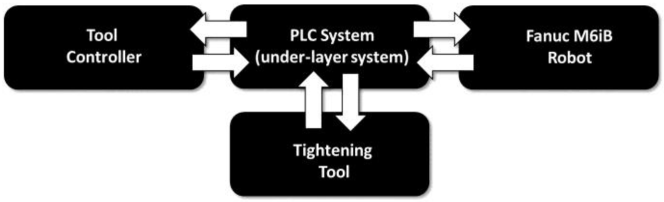

Fig. 4. Implementation of robot and tool control set-up, based on Beckhoff TwinCAT software system and PCs network connected through Ethernet protocol.

B. Neural network rational

A model free approach inherently incorporating these dynamics is proposed: a set of neural networks employing experimental measurements of torque, angular displacement, and clamping force as extracted from fastening M24 bolts and nuts is proposed.

Ten experimental trials are performed at maximum speed of the tightening tool - i.e. maximum voltage of the control input signal (± 10 V, [15]) - Fig. 3. The maximum speed and torque applied are 190 rpm and 140 N·m, respectively. Angular displacement of the tool (α), applied tightening torque (τ), and clamping force (F_{clamp}) are measured. Data are sampled at 2 kHz (i.e. with a real-time cycle time of 500 μs) with a Windows Control and Automation Technology Beckhoff TwinCAT 3 system [15, 38]; the system is coupled with the tool and sensors via an industrial Personal Computer (PC) and a desktop PC; the two PCs are connected through an Ethernet communication protocol; a software architecture integrates models and programs which are developed under Matlab and Simulink environment (by Mathworks Inc.) and then compiled into real-time operating code and executed on an industrial PC by the Beckhoff TwinCAT software system [38] (Fig. 4).

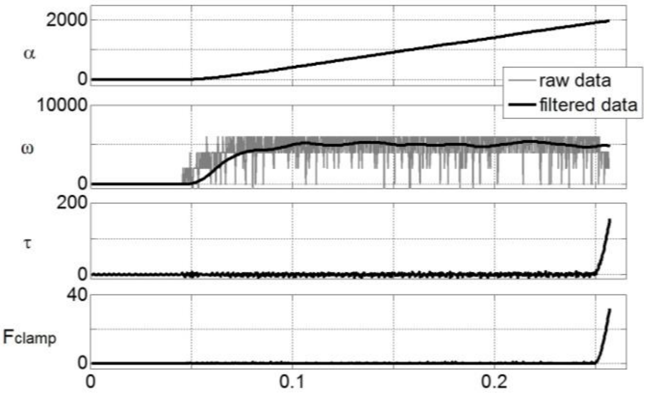

Fig. 5. From top to bottom panels, representative data of trial n. 1 (out of 10): tool angular displacement (α [°]), raw and filtered angular speed (ω [°/s]) - in grey and black color, respectively -, tightening torque (τ [N·m]) and clamping force (F_{clamp} [kN]).

Since dynamics equation of the tightening would the initial conditions of the process [39], it makes sense to introduce a further parameter in the neural network controller structure, namely the derivative of the tool angular displacement or angular speed (ω). This signal can be inferred as the derivative of the angular position of the tool in such a way that the network is fed with the position of the tool and its speed; a third order elliptic filter, with a cut-off frequency of 10 Hz, 0.1 dB pass-band ripple, and stop-band at −100 dB, is applied in order to smooth the noise of the angular position derivative. The calculation of the derivative and the filtering are implemented and off-line applied with the Matlab Programming Language.

Fig. 5 shows the final set-up of one experimental acquisition (trial n. 1), including the aforementioned rotational speed of the tool: from the top panel to the bottom one, the angular displacement, angular speed (raw and filtered signals), tightening torque and clamping force are reported, respectively. The figure is well representative of all the experimental data collected, which show similar patterns in the other 9 trials; Table I reports the mean and two times standard deviations, as well as the percentage of the standard deviation compared with the mean, of α, ω, τ, F_{clamp}, respectively, of all trials. According to these values, the repeatability of the tightening processes is appropriate, with an average execution time of 0.52 ± 0.02 s, an angular variability of less than 3% (i.e. less than 10% of the number of turns required to accomplish full tightening, namely 1975°/360° ≅ 13 turns). Similarly, the variability of torque, angular speed and clamping force are less than 11%, 5% and 16%, respectively.

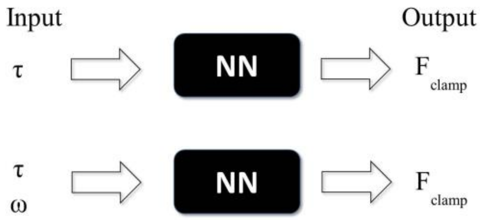

Fig. 6. I/O neural network configuration with tightening torque (τ) and angular speed (ω) inputs - single or double configurations, top and bottom scheme, respectively - and clamping force (F_{clamp}) output.

Despite such a good repeatability of the data over all trials, a normalization process of the measurements is needed to in order to properly input the neural networks (see Section II, par. D). Beforehand we need to define the structure of the networks.

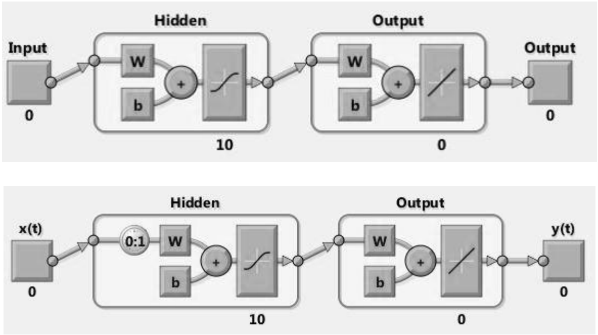

Fig. 7. Fitting (static) and time series (dynamic) neural network set-up (top and bottom panels, respectively): hidden and outer layer contains 10 and 1 neurons, embedding sigmoid and linear transfer functions, respectively.

C. Design and training of the neural networks

A set of neural networks are adopted to model the clamping force (F_{clamp}) as the output of the networks; on the other side, the tightening torque (τ) and the derivative of the tool angular displacement (i.e. the angular speed of the tool, ω) are applied as inputs of the networks.

Two network configurations, with one and two inputs, respectively, are implemented with the Matlab Programming Language (Fig. 6). For each one of these configurations, two types of network are defined: static or fitting network and dynamic or time series network. Therefore four types of neural networks are employed, according to a combination of the different set of inputs (i.e. one or two input) and types of network (i.e. static or dynamic network).

TABLE II
TRAINING PARAMETERS

parameter	*value*	*description*
N_{max}	1000	*n. of training epochs (stop condition)*
MSE_G	0	*MSE goal (stop condition)*
F_{max}	6	*n. of consecutive epochs where training performance fails to improve (stop condition)*
$\mu_{initial}$	0.001	*initial learning rate*
μ_{max}	1e10	*maximum learning rate*
$\mu_{decreasing}$	0.1	*decreasing decay rate*
$\mu_{increasing}$	10	*increasing decay rate*

The static and dynamic networks are both feedforward neural networks fitting the input-output model. The dynamic model applies a time delay associated with the input [40]. Both types of networks embed two layers of neurons, namely an hidden and output layer. The former is populated with 10 neurons, each one using a sigmoid transfer function. A linear transfer function is applied to the latter one, which consists of a single neuron. Fig. 7 shows the configurations of both neural networks, where '*w*' and '*b*' refer to the weight and bias of each neuron, respectively [40].

Sensor data are pre-processed, i.e. normalized, and post-processed, i.e. de-normalized, with respect to their maximum and minimum values (details in Section II, par. C). To optimize the learning of the system, data are randomly divided into 3 groups: the training phase of each network is performed with 70% of data (group 1), whereas validation and testing are implemented with 15% of data each (group 2 and 3). Data of the second group are used to validate the network generalization capability and stop the training phase in case of an over-fitting detection. On the other side, the data of the third group are applied to the trained network in order to validate its generalization performance vs. a novel data set, irrespective of the training performance.

The training is accomplished with a Levenberg-Marquardt back-propagation method [41]; the adaptive learning parameters are set-up as it is reported in Table II; a momentum term, equal to 0.9, is also included to stabilize it. Weights and biases are initialized with a Nguyen-Widrow function and then updated after presenting all training set to the network [42].

Furthermore, in order to improve the training performance and avoid local minima, multiple training runs are performed with each network configuration, always commencing the process from a randomly distributed set of initial weights.

Mean Squared Error (MSE) is used to monitor the training performance, namely:

$$\text{MSE} = \frac{1}{n}\sum_{1}^{n}(\widehat{F}_{clamp_i} - F_{clamp_i})^2 \qquad (1)$$

where n is the numbers of experiments of the training set, F_{clamp_i} is the effective clamping force of the *i-th* experiment and $\widehat{F}_{clamp_i}$ the value as predicted from the network [40].

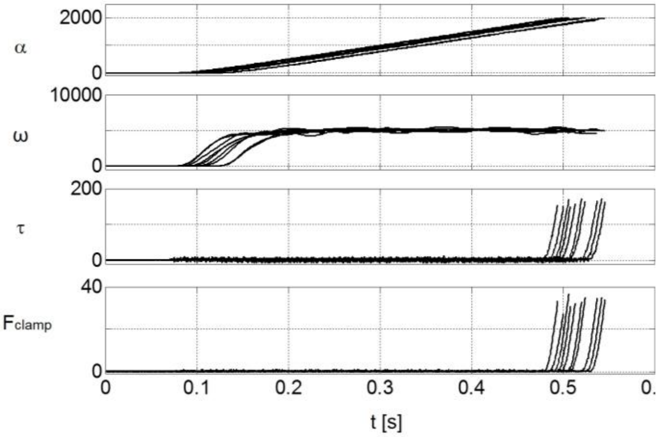

Fig. 8. Normalization of experimental data with null threshold of the tightening torque ($\tau_{trigger}$ = 0 N·m).

D. Data normalization

A proper normalization should not dissipate the information inherited from the data and, at the same time, should conform with the required range of values of the I/O transfer functions of the network. To this aim, a two stage normalization is performed.

Stage 1: clamping force (F_{clamp}) and torque (τ) values are approximately null during most of the tightening process; for instance, during the 1st trial, this happens between 0 and 0.48 s (Fig. 5). As a consequence, this part of not informative experimental data can be removed from the training set; a new initial time (t_{inil}^{i}) is defined for each of the *i-th* trial, which is initialized and triggered as a specific threshold of torque is reached ($\tau_{trigger}$). Four different triggering values are adopted, namely 0, 5, 10 and 15 N·m. A separate training, according to details reported in Section II, part C, is performed for each one of the 4 training set as obtained by applying the 4 threshold values. Results of this normalization stage, as performed with null value of threshold ($\tau_{trigger}$ = 0 N·m), are shown in Fig. 8.

Stage 2: Regardless of the high repeatability of data (Section II, part B and Table I), a further normalization stage is introduced to improve data homogeneity and to cluster all experimental acquisitions into a single data set. Since each experiment takes different execution time, a normalized time (t_{norm}^{i}) is defined for each *i-th* trial (*i* = 1,..,10) with respect to the effective execution time (t_f^{i}); therefore it holds:

$$t_{norm}^{i} = t / t_f^{i}, \text{ where } t_{norm}^{i} \in [0,1] \text{ s} \qquad (2)$$

Finally, the network training is performed after processing the data, according to the following steps:

- data of the ten experiments are loaded
- time array of the *i-th* experiment is overwrite from *t* = 0 s to t_f^{i} at a sampling frequency of 2 kHz (Section II, part B) $\tau_{trigger}$ threshold is set and values of angular displacement (α) and execution time are reset to null value in correspondence of the triggering point
- angular speed (ω) is inferred from the novel α vector and then filtered (Section II, part B)
- parameters are normalized with respect to t_{norm}^{i} from 0 to 1 s
- patterns are interpolated to preserve same length and dimension of each array over all trials, irrespective of their effective execution time (t_f^{i}).

TABLE III
MSE OF STATIC NEURAL NETWORK, $\tau_{TRIGGER}$ = 0 N M

	input			
	τ		*τ, ω*	
trials	*train*	*test*	*train*	*test*
1	4.04e+04	6.38e+04	4.25e+04	3.42e+04
2	2.66e+04	6.46e+04	3.99e+04	1.65e+05
3	3.10e+04	4.66e+05	5.56e+04	3.38e+04
4	1.51e+05	7.54e+04	4.07e+04	1.37e+05
5	1.78e+05	1.27e+05	3.63e+04	1.04e+05
6	4.59e+04	1.26e+05	6.19e+04	6.24e+04
7	3.36e+04	1.56e+05	8.60e+04	5.61e+04
8	2.90e+04	8.68e+04	3.73e+04	2.50e+04
9	7.60e+04	3.06e+05	4.87e+04	6.60e+04
10	6.18e+05	2.16e+05	1.04e+05	2.80e+05

III. RESULTS

For each network configuration, ten training sessions are performed: fitting and time series networks, as well as single and double inputs are used (Figs 6-7; Section II, part B). Networks are trained with normalized experimental data (Section II, part D) according to protocol reported in Section II, part C.

TABLE IV
MSE OF DYNAMIC NEURAL NETWORK, $\tau_{TRIGGER}$ = 0 N M

	input			
	τ		*τ, ω*	
trials	*train*	*test*	*train*	*test*
1	5.56e+04	1.12e+04	4.55e+04	2.16e+04
2	3.67e+06	2.34e+07	3.95e+04	5.47e+06
3	2.98e+04	2.59e+04	4.08e+04	7.72e+06
4	2.97e+04	1.63e+04	1.77e+04	1.27e+06
5	2.77e+05	5.89e+04	4.37e+04	1.62e+04
6	2.07e+04	3.65e+07	5.62e+04	2.66e+04
7	4.02e+04	1.65e+04	6.20e+05	5.53e+05
8	2.23e+04	2.06e+05	6.78e+04	2.14e+04
9	4.51e+04	9.92e+04	3.41e+04	2.10e+05
10	1.82e+04	3.88e+09	4.33e+05	5.05e+05

Results are summarized in Tables III-V, which report the training and testing MSE (*train* and *test* columns, respectively), according to the inputs of the networks - torque (τ) and torque and angular speed (τ and ω) –, their configurations - fitting and time series – and the value of the triggering torque - $\tau_{trigger}$ = 0, 5, 10, 15 N·m.

Particularly, Table III refers to results obtained with static networks and null $\tau_{trigger}$, whereas Table IV reports results achieved with same set of parameters but time series networks.

Each table shows the outcomes of 10 consecutive training processes, according to the ten random initializations (Section II, part C).

TABLE V
MSE MEAN & STD
OF STATIC & DYNAMIC NETWORKS, $\tau_{TRIGGER}$ = 5, 10, 15 N M

τ input					
$\tau_{trigger}$ [N·m]	static (s) & dynamic (d)	train		test	
		mean	std	mean	std
5	s	1.07e+05	1.09e+05	2.65e+04	2.07e+05
	d	1.18e+06	3.36e+06	6.50e+07	2.04e+08
10	s	2.94e+05	1.57e+04	3.65e+05	1.09e+05
	d	2.88e+05	4.39e+04	3.31e+05	6.15e+04
15	s	7.65e+05	2.95e+04	8.61e+05	9.41e+04
	d	7.54e+05	4.78e+04	8.19e+05	1.39e+05
τ, ω input					
5	s	5.73e+04	1.45e+04	1.21e+05	6.56e+04
	d	1.47e+05	1.83e+05	4.64e+05	9.97e+05
10	s	2.63e+05	5.05e+04	3.08e+05	9.44e+04
	d	2.32e+05	3.76e+04	2.72e+05	7.13e+04
15	s	5.37e+05	5.60e+04	5.65e+05	9.26e+04
	d	5.14e+05	5.95e+04	5.73e+05	7.10e+04

On average, mean and standard deviation of training and testing MSE of the static network with null $\tau_{trigger}$ and single input τ, are $(1.23 \pm 1.82)\cdot 10^5$ and $(1.69 \pm 1.29)\cdot 10^5$, respectively, whereas double inputs (τ and ω) leaded to $(0.52 \pm 0.23)\cdot 10^5$ and $(0.96 \pm 0.79)\cdot 10^5$, respectively.

A complete set of the average and standard deviation of training and testing MSE, for the different configurations is reported in Table V.

Bar distribution of training MSE mean and standard deviation is shown in Fig. 9 for all network configurations, namely for static and dynamic networks (*s* and *d* within the figure, respectively), with single and double inputs (in black and grey color, respectively) and with different performed normalization ($\tau_{trigger}$ = 0, 5, 10, 15 N·m).

Similarly, testing performances are reported in Fig. 10, which shows mean and standard deviation of the testing MSE error according to the same set of network configurations.

IV. DISCUSSION

A qualitative and quantitative interpretation of results is given in part A. Effect of the data pre-processing on networks performance are analyzed in part B. Finally, generalization properties of the networks are discussed, as well as results vs. previous literature.

A. Results interpretation

To correctly interpret the physical meaning of the training and testing MSE, definition of the error has to be taken into account (eq. (1), Section II, part C): namely, an MSE value of 10^5 N^2 is equivalent to an effective error of ± 316 N. Since average value of F_{clamp} is 33.13 kN (Table I), the aforementioned error is less than 0.96% of the final clamping force (i.e. ≈ 31.60 kg vs. 3313.00 kg).

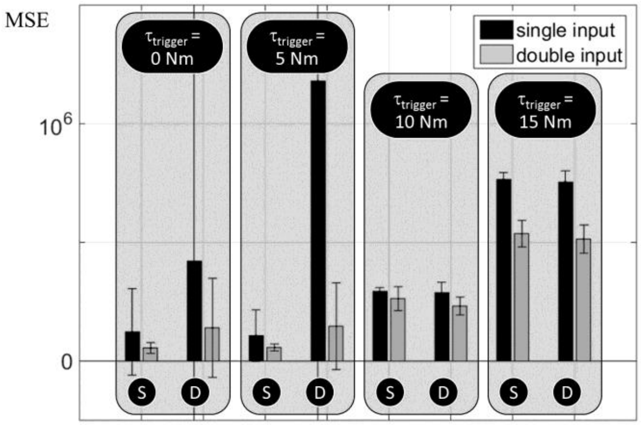

Fig. 9. Mean and standard deviation of *training* MSE of 10 learning processes vs. different static (*s*) and dynamic (*d*) networks' configurations with diverse triggering torque ($\tau_{trigger}$ = 0, 5, 10, 15 N·m).

According to this estimation and results (Figs 9-10 and Tables III-V), networks with double inputs are qualitatively better performing than those ones having same configuration but single input. Normalizing data with small $\tau_{trigger}$ values, i.e. 5 N·m and 10 N·m, makes static networks outperforming the dynamic ones, whereas, when data are appropriately clustered - i.e. $\tau_{trigger}$ grows at 15 N·m – performance of dynamic networks become comparable with those of the static ones. Therefore, to simultaneously (a) optimize network performance and (b) increase the informative content of data, (c) while keeping network configuration as much simple as possible (i.e. static), the value of the triggering torque should be maintained low. However, even with high value of torque threshold, errors do not dramatically increase (Tables III-V).

Being *p* the level of significance, paired T-test shows that there is no statistically significant difference between training single and double input networks with $\tau_{trigger}$ equal to 0 N·m (p = 0.24) [1]. With the same value of threshold torque, there is also no significant difference between training static or dynamic networks having single (p = 0.45) and double inputs (p = 0.19). A similar behavior is preserved when value of $\tau_{trigger}$ is increased to 5 N·m, either when comparing networks with single or double inputs, or comparing static and dynamic networks having the same input configuration. By increasing the $\tau_{trigger}$ value to 10 N·m, this not significant difference between static and dynamic configurations lasts, whereas a substantial difference is observed when comparing the training of static and dynamic networks having double inputs (p = 0.03). By further increasing threshold torque value to 15 N·m, this latter difference disappears; nevertheless, using single or

[1] We assume that the difference is significant if p is equal or smaller than 0.05, which means a 5% or lower probability of being wrong in asserting that two sets of data are different.

double inputs, either with static or dynamic networks, is making a significant difference (p = 8.94e-07 and p = 1.34e-05, respectively).

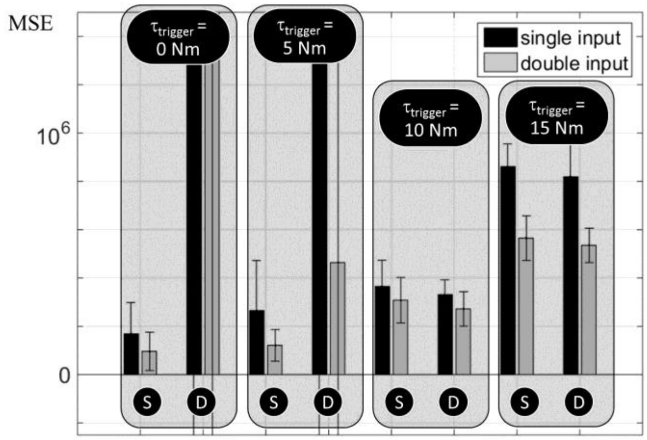

Fig. 10. Mean and standard deviation of *testing* MSE of 10 learning processes vs. different static (s) and dynamic (d) networks' configurations with diverse triggering torque ($\tau_{trigger}$ = 0, 5, 10, 15 N·m).

B. Effect of data pre-processing

Results are also depending on pre-processing of raw data. Looking at experimental data and at the first stage of the normalization process (Section II, part D), time patterns of force and torque are close to be null during the majority of time of each trial: therefore, there is no substantial knowledge which is returned to the network during this experimental phase; conversely, in terms of learning, the most useful stage of the trial is when force and torque start increasing, which typically occurs between t equal to 0.48 s and 0.53 s (Fig. 5). This trend suggests to endorse data pre-processing with an higher values of the torque threshold (for instance, $\tau_{trigger}$ = 10N·m or 15 N·m) to avoid overloading the networks with data containing null information. The handicap of selecting high values of $\tau_{trigger}$ is a strong reduction of data size training the networks: in fact, despite the high sampling frequency (Section II, part B), the tightening process is inherently fast and the number of experimental data may be strongly reduced because of the selection of an high $\tau_{trigger}$ value (see Fig. 5 and 8, which display time patterns of the experimental data before and after the normalization procedure, respectively). As an immediate consequence, performing networks training with few experimental acquisitions may affect learning performance, since the dimension of the training set has to be modulated according to the size of the network (i.e. its dimension and number of neurons) [43-46].

Focusing on the second stage of the pre-processing, i.e. the time normalization (Section II, part D), this phase also introduces some side effects: novel offsets between data patterns of different experiments may be not originally present in the raw data since applied thresholds of torque may have introduced time shift of the starting point of each trial acquisition. To overcome these drawbacks, the beginning of each experimental data set may be defined with another criteria which is based, for example, on the signal derivative – e.g. the change of slope of the angle-torque curve. Such an approach has been suggested by Shoberg [30] and effectively this transition of the signal feature represents the physical 'alignment' of the bolt and nut threads by detecting the beginning of the proper tightening.

C. Learning performance

Generalization skill of the networks can be extrapolated from comparing testing and training MSE (Tables III-V): on average testing error is higher than training one, suggesting a potential over fitting [45, 46]. Therefore, new set of networks may be considered in a further step of this research, to explore networks performance vs. numbers of layers and neurons, and finally optimize their structure vs. the size of the data set [45].

Finally, to validate networks outcome, a trained static network with double inputs and threshold torque equal of 0 N·m is tested: assuming a constant angular speed of tool (ω = 5000 °/s) and a linear tightening torque profile (τ = 0 ÷ 160 N·m), a simulated tightening process is implemented (Fig. 11). Feeding the network with this data set of input leads to a set of output which is compared with the experimental data of th e10 trials (Fig. 11): a 3D representation can be displayed, where signals lying on the x, y plane refer to the initial stage of the tightening process, i.e. when the bolt is screwed on the threads of the nut without applying any force (null values in the z-axis). Fig. 11 outlines the proper fitting between the network output, which is reported in black color, and the overall trends of experiments (shown in grey color).

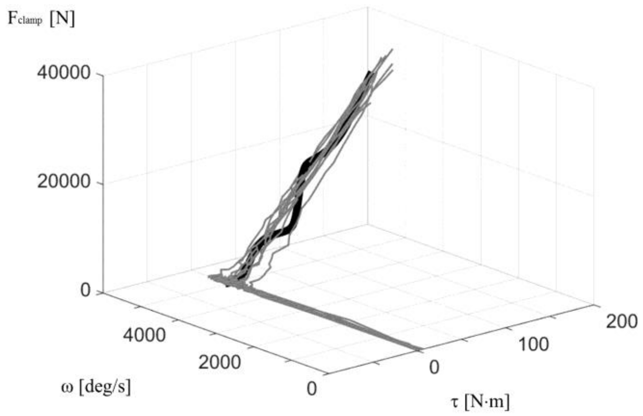

Fig. 11. Three dimensional representation of data from 10 tightening trials (grey color) vs. neural network output (black color): tightening torque (τ, x-axis), angular speed (ω, y-axis) and resultant clamping force (F_{clamp}, z-axis) are reported vs. output simulation of static network with double input and $\tau_{trigger}$ equal to 0 N·m (see details in the text).

In previous work, Fujinaka et al. [31] proposed a combination of neural network controllers for bolt tightening with an impact wrench pneumatically actuated. Controllers were able to perform materials classification in between the bolt and nut as well as to output achieved clamping force. Nevertheless, this latter force estimation was performed in correspondence of specific angular positions values, and precisely in the *snug point*, which triggers when force proportionally increases with angle. Conversely, rather than performing single punctual estimations, the proposed set-up of this work allows modeling the entire time history of the clamping force occurring during the tightening process, from bolt and nut alignment, passing through partial and full engagement of threads, to final tightening [15],.

Another approach, which was based on Fuzzy Logic control, was proposed from Dhayagude et al. in [33]: here, screw fastening was supervised to prevent process failure (cross-threading, screw jamming, etc.) and to obtain a precise tightening torque. Performance of this controller were validated through model simulations based on fastening dynamics and Eulero-Lagrange approach; in this context, it can be notice that the neural network architecture which is presented in this paper inherently incorporates this dynamic, namely the bolt-nut plant, since it fits related experimental data and has been validated by employing these latter measurements.

V. Conclusion

A set of neural networks for the modeling of the clamping force on tightening M24 bolts and nuts in wind turbine assembly has been presented.

Experimental acquisitions have been performed with hardware and software set-up: a tightening instrumented tool, a washer sensor and a Beckhoff TwinCAT software architecture running at 2 kHz within an industrial PC; a Fanuc M6iB robot and a Matlab/Simulink environment running in a second PC are wired connected through Ethernet protocols. Experimental data are normalized in time and with respect to 3 triggering thresholds of torque to train and test the networks whose number of neurons has been kept constant; in line with the dynamical characteristics of the process, two sets of inputs have been considered, which supply the network with the needed parameter and initial conditions: in this context, derivative of tool angular speed is introduced as one of the network input.

Results have shown that this set-up is capable of modeling the time patterns of clamping force from tool tightening torque and angular speed.

A further step of this study may emphasize optimization and adaptation of networks structure and size vs. dimensions of the experimental data sets [45]. Another critical aspect is the optimization of the data pre-processing and normalization; this latter phase may focus on identifying invariants or torque-angle signatures: some of them have been already identified, like, for instance, the rundown phase (i.e. the prevailing phase of the tightening occurring at the beginning of the process, when the clamping force is almost zero and the only resistance is due to friction in the threaded region [28, 35]), the alignment, elastic clamping and post yield zones, as reported from Drumheller in [35].

References

[1] G.M. Joselin Herbert, “A review of wind energy technologies,” *Renewable and Sustainable Energy Reviews*, vol. 11, pp. 1117-1145, 2007.

[2] C.S. Ezio, “Exploitation of wind as an energy source to meet the worlds electricity demand,” *Wind Eng*, vols. 74-76, pp. 375-387, 1998.

[3] R. Saidur, “A review on global wind energy policy,” *Renewable and Sustainable Energy Reviews*, vol. 14, pp. 1744-1762, 2010.

[4] Y. Lin, L. Tu, H. Liu, W. Li, “Hybrid Power Transmission Technology in a Wind Turbine Generation System,” *IEEE/ASME Transactions on Mechatronics*, vol. pp, no. 99, pp. 1-8, 2014.

[5] “Energy Siemens,” Siemens Corp. [Online]. Available: http://www.energy.siemens.com/mx/pool/hq/power-generation/renewables/wind-power/6_MW_Brochure_Jan.2012.pdf [Accessed 27 06 2014].

[6] J. Nilsson and L. Bertling, “Maintenance management of wind power systems using condition monitoring systems—Life cycle cost analysis for two case studies,” *IEEE Trans. Energy Convers*., vol. 22, no. 1, pp. 223–229, Mar. 2007.

[7] A. Ozdemir, P. Seiler, G.J. Balas, “Wind Turbine Fault Detection Using Counter-Based Residual Thresholding”, 18th IFAC World Congress, 2011.

[8] W. He, S.S. Ge, “Vibration Control of a Nonuniform Wind Turbine Tower via Disturbance Observer,” *IEEE/ASME Transactions on Mechatronics*, vol. pp, no. 99, pp.1-8, 2014.

[9] M. Sharpe, “Robotic fabrication of wind turbine power generators,” *Welding Journal*, vol. 8, pp. 41-44, 2009.

[10] “ABB robotics makes wind turbine manufacturing more efficient,” ABB Group. [Online]. Available: http://www.abb.co.uk/cawp/seitp202/50a263c7fa5d15c5c12577ff0044d2e9.aspx [Accessed 28 07 2014].

[11] “An automated approach to blade manufacturing,” Wind Systems. [Online]. Available: http://www.windsystemsmag.com/article/detail/76/an-automated-approach-to-blade-manufacturing- [accessed 28 07 2014].

[12] “Wind turbine manufacturing planning,” Dassault Systems. [Online]. Available: http://www2.3ds.com/fileadmin/PRODUCTS/DELMIA/PDF/EPU_Wind_Turbine_Mfg_Plan_Flyer.pdf [Accessed 28 07 2014].

[13] N. Karnati, B.A. Kent, E.D. Engeberg, “Bioinspired Sinusoidal Finger Joint Synergies for a Dexterous Robotic Hand to Screw and Unscrew Objects with Different Diameters,” *IEEE/ASME Trans. On Mechatronics*, vol. 18, no. 2, 2013.

[14] X. Wu, Y. Li, F. Li, Z. Yang, W. Teng, “Adaptive Estimation-Based Leakage Detection for a Wind Turbine Hydraulic Pitching System,” IEEE/ASME Transactions on Mechatronics, vol. 17, no. 5, pp. 907-914, 2012.

[15] C. Deters, H.K. Lam, E.L. Secco, H.A. Würdemann, K. Althoefer, “Accurate bolt tightening using model-free fuzzy control for wind turbine hub bearing assembly,” *IEEE Transactions on Control Systems Technology*, vol. 99, 2014.

[16] S.A. Nassar, G.C. Barber, D. Zuo, “Bearing friction torque in bolted joints,” *Tribology Transactions*, vol. 48, 69-75, 2005.

[17] Y. Maeda and M. Iwasaki, “Initial friction compensation using rheology based rolling friction model in fast and precise positioning,” *IEEE Trans. Ind. Electron.*, vol. 60, no. 9, pp. 3865–3876, Sep. 2013.

[18] J. H. J. Potgieter and M. J. Kamper, “Torque and voltage quality in design optimization of low-cost non-overlap single layer winding permanent magnet wind generator,” *IEEE Trans. Ind. Electron.*, vol. 59, no. 5, pp. 2147–2156, May 2012.

[19] H. Chaoui and P. Sicard, “Adaptive fuzzy logic control of permanent magnet synchronous machines with nonlinear friction,” *IEEE Trans. Ind. Electron.*, vol. 59, no. 2, pp. 1123–1133, Feb. 2012.

[20] T. Yokoyama, M. Olsson, S. Izumi, and S. Sakai, “Investigation into the self-loosening behavior of bolted joint subjected to rotational loading,” *Eng. Failure Anal.*, vol. 23, pp. 35–43, Jul. 2012.

[21] H. A. Talebi and K. Khorasani, “A neural network-based multiplicative actuator fault detection and isolation of nonlinear systems,” *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 3, pp. 842–851, 2013.

[22] D. A. Dirksz and J. M. A. Scherpen, “Power-based set point control: Experimental results on a planar manipulator,” *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 5, pp. 1384–1391, Sep. 2012.

[23] “BolTight hydraulic bolt tensioning,” Boltight Echometer, http://www.echobolt.com/wp-content/themes/twentyten/ppt/Echometer_Overview1.pdf [Accessed 27 06 2014].

[24] “Bolt-tightening handbook”, Catalogue n. TSI 1101 AE, April 2001,The SKF Group.

[25] L.D. Seneviratne, F.A. Ngemoh, S.W.E. Earles, K.A. Althoefer, “Theoretical modelling of the self-tapping screw fastening process,” Proc. Inst. Mech. Eng., C, *J. Mech. Eng. Sci.*, vol. 215, no. 2, pp. 135–154, 2001.

[26] K. Althoefer, B. Lara, Y.H. Zweiri, L.D. Seneviratne, “Automated failure classification for assembly with self-tapping threaded fastening using artificial neural networks,” Proc. Inst. Mech. Eng., C, J. Mech. Eng. Sci., vol. 222, no. 6, pp. 1081–1095, 2008.

[27] M. Klingajay, L.D. Seneviratne, K. Althoefer, “Identification of threaded fastening parameters using the Newton Raphson method,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Oct. 2003, pp. 2055–2060.

[28] C. Deters, E.L. Secco, H.A. Würdemann, H.K. Lam, L.D. Seneviratne, K. Althoefer, “Model-free Fuzzy Tightening Control for Bolt/Nut Joint Connections of Wind Turbine Hubs,” IEEE International Conference on Robotics and Automation (ICRA), 270-276, 2013.

[29] H.K. Lam, H. Li, C. Deters, E.L. Secco, H.A. Würdemann, K. Althoefer, “Control design for interval type-2 fuzzy systems under imperfect premise matching”, *IEEE Transactions on Industrial Electronics*, vol. 61, no. 2, pp. 956-968, 2014.

[30] R.S. Shoberg, “Tightening Strategies for Bolted Joints,” 11th Annual Tech Conf on Fastening Technology, 1999.

[31] T. Fujinaka, H. Nakano, S. Omatu, “Bolt tightening control using neural networks,” IEEE International Conference on Systems, Man, and Cybernetics, vol. 3, 1390-1395, 2001.

[32] S. Izumi, T. Yokoyama, A. Iwasaki, S. Sakai, “Three-dimensional finite element analysis of tightening and loosening mechanism of threaded fastener,” *Engineering Failure Analysis*, vol. 12, 604-615, 2005.

[33] N. Dhayagude, Z. Gao, F. Mrad, “Fuzzy logic control of automated screw fastening,” *Robotics and Computer-Integrated Manufacturing*, vol. 12, no. 3, pp. 235-242, 1996.

[34] J.H. Bickford, S.A. Nassar, Handbook of bolts and bolted joints, Marcel Dekker, New York, 1998.

[35] J. Drumheller, “Understanding torque-angle signature of bolted joints, fastener testing engineer,” Threaded Fastener Torque-Angle Curve Analysis [Online]. Available: http://www.pcbloadtorque.com/pdfs/technicalArticles/UnderstandingTorqueAngleSignatures.pdf [Accessed 27 06 2014].

[36] S.A. Nassar, X. Yang, “Torque-angle formulation of threaded fastener tightening,” *Journal of Mechanical Design*, vol. 130, 1-4, 2008.

[37] “Coefficients of friction,” Roymech Co. [Online]. Available: http://www.roymech.co.uk/Useful_Tables/Tribology/co_of_frict.htm [Accessed 30 06 2014].

[38] "Beckhoff (TwinCAT 3)," Beckhoff GmbH, 01 02 2012. [Online]. Available: www.beckhoff.de [Accessed 01 02 2013].

[39] E.L. Secco, G. Magenes G., “A feed-forward neural network controlling the movement of a 3 d.o.f. finger,” IEEE Transactions on Systems, Man, and Cybernetics Part A, vol. 32 (3), 437-445, 2002.

[40] “Matlab Help,” The Mathworks Inc. [Online]. Available: http://www.mathworks.co.uk/help/nnet/ref/mse.html?searchHighlight=mse [Accessed 27 06 2014].

[41] Secco E.L., “Movement Control of a 3 d.o.f. artificial finger: dynamic learning and execution of the natural movment, “ Phd Thesis, University of Pavia, Italy, 2001.

[42] D. Nguyen, B. Widrow, “Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights, ” Proc. of the International Joint Conference on Neural Networks, 3:21–26, 1990.

[43] E.L. Secco, G. Magenes, “A Life-Like Control Algorithm for the Natural Movement of a 3 dof Finger,” Proc. Int. Joint Conf. BMES EMBS, Houston, pp. 2375-2376, 2002.

[44] G. Magenes, S. Ramat, E.L. Secco, “Life-like Sensorimotor Control: from Biological Systems to Artifacts,” *Current Psychology of Cognition*, Vol. 21(4-5), pp. 565-596, 2002.

[45] S. Haykin, Neural Networks—A Comprehensive Foundation, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[46] R. Rojas, Neural Networks: A Systematic Introduction. New York: Springer-Verlag, 1996.

.