Machine Learning for Active Gravity Compensation in Robotics: Application to Neurological Rehabilitation Systems

Axier Ugartemendia[1,2], Daniel Rosquete[1,2], Jorge Juan Gil[2], Iñaki Díaz[1,2] and Diego Borro[1,2]

***Abstract*— Robotic rehabilitation for post-stroke therapies is actually an emerging new domain of application for robotics with proven success stories and clinical studies. New robotic devices and software applications are hitting the market with the aim of assisting specialists carrying out physical therapies and even allowing patients exercising at home. Rehabilitation robots are designed to assist patients performing repetitive movements with their hemiparetic limbs to regain motion. A successful robotic device for rehabilitation demands high workspace and force feedback capabilities similar to a human physiotherapist. These desired features are usually achieved at the expense of other important requirements such as transparency and backdrivability, degrading the overall human-machine interaction experience. We present an active gravity compensation method that can highly improve the performance of mechatronic systems used for rehabilitation and many other domains of robotic applications. Traditional algorithms to obtain active gravity compensation usually require the static equilibrium equations of the system. However, for complex mechatronic configurations, solving these equations is not straightforward. The use of Machine Learning methods can achieve gravity compensation without the need to solve the equilibrium equations. To validate the performance of the proposed approach, HomeRehab robotic rehabilitation system is used to obtain experimental results.**

I. ROBOTIC REHABILITATION

Stroke is currently the second most frequent cause of death after coronary artery disease and its prevalence is increasing at an alarming rate. Hemiparesis is the most common outcome of stroke leading to movement deficiency. Fortunately, rehabilitation can help hemiparetic patients to learn new ways of using and moving their weak arms and legs. It is also possible with immediate therapy that people who suffer from hemiparesis may eventually regain movement. Although there are several approaches, extensive task specific repetitive movement is one of the safest and most effective methods to regain lost mobility of the affected limbs. This therapy requires incessant medical care and intensive rehabilitation often requiring one-on-one manual interaction with the physical therapist.

Robotic rehabilitation is an emerging field that allows a patient performing exercises with the assistance of a robotic device [1]. These systems can be used in providing therapy (even at the patient's home) for a long period of time irrespective of skills and fatigue compared to manual therapy. Besides the cost-effective aspect, robotic devices introduce higher accuracy and repeatability in performing exercises. Precise measurements of quantitative parameters by means of robotic instrumentation also improve objective monitoring of patients recovery. Furthermore, rehabilitation robots can be combined with virtual reality environments to engage and push patients to keep training as the patient's motivation and cognitive involvement has a great impact on the outcome of rehabilitation [2].

Currently, there are several devices in the market that give a robotic solution to these repetitive movements, and have been installed in many hospitals around the world. Some examples are: InMotion ARM (Bionik Labs), ReoGo (Motorika) and Armeo Power (Hocoma). For a successful rehabilitation system, the robot must have the capability to deliver physical forces similar to manual therapy. Mechanically, this implies developing robots with high workspace and force feedback features. Such systems have in turn the drawback of being bulky and heavy degrading final interaction experience with the patient. Among the different technical challenges of these systems, gravity compensation is key for high rehabilitation performance.

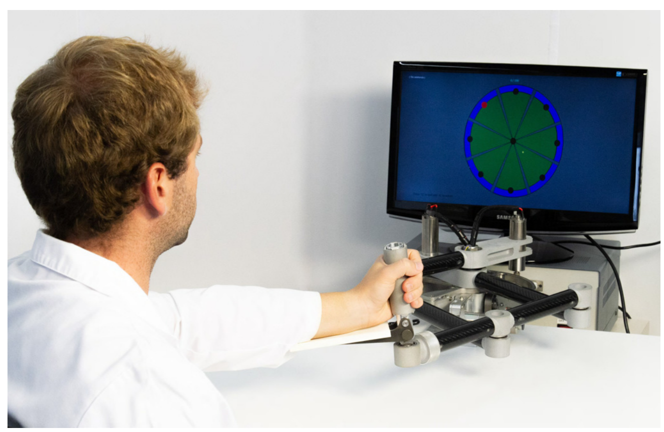

Fig. 1. HomeRehab robotic rehabilitation system developed at Ceit

In previous work [3], we have developed the HomeRehab robotic system capable of restoring haptic effects at its handle for upper limb rehabilitation (Fig. 1). This mechanism has three degrees of freedom and it consists of a pantograph that pivots on a horizontal axis, but it is not perfectly gravity-balanced. Thus, an active compensation strategy is needed. Note that in this work gravity compensation means removing the gravity components of the mechanical device in order to

[1]Axier Ugartemendia, Daniel Rosquete, Iñaki Díaz and Diego Borro are with the Materials and Manufacturing Division of CEIT-Basque Research and Technology Alliance (BRTA), 20018 Donostia-San Sebastián, Spain. {augartemendia, drosquete, idiaz, dborro} @ceit.es

[2]Jorge Juan Gil is with the Department of Mechanical Engineering and Materials, Tecnun, University of Navarra, 20018 Donostia-San Sebastián, Spain. jjgil@tecnun.es

Manuscript 3074 submitted to 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received May 13, 2020.

IEEE Robotics and Automation Magazine (RAM) paper, presented at IROS 2020. It should be cited as a RAM paper.

make it transparent or imperceptible to the patient, in the sense that during the rehabilitation exercises the patient only has to overcome the weight of his/her own arm.

II. RELATED WORK

Gravity compensation is a basic need in robotics, and more specifically in haptics, to ensure system usability and transparency. It is also a key specification of mechatronic rehabilitation devices [4], so that the users can move their arm freely without feeling (and holding) the weight of the robot during the therapy. Besides, friction could be an additional limiting factor of these mechanisms, and some strategies try to compensate both unknown static friction and gravity forces [5]. A comparison of different algorithms for gravity compensation of parallel mechanisms can be found in [6].

In the field of rehabilitation devices, several gravity compensation strategies have been applied that are also valid for any mechatronic system. In some cases, the mechanism is designed to be gravity-balanced, that is, to be in neutral equilibrium without requiring joint actuator torques [7]. This feature can also be achieved by adding a passive mechanism to the robotic arm [8]. These solutions are intrinsically safe, but they are difficult to design.

A. Analytical Approaches

The analytical active compensation methods balance the system acting on each joint according to a gravity model of the mechanism. In this model, the torques depend on the pose and the weight distribution of the links. The positions of the centers of mass and the weights of the links could be estimated from the CAD model. However, these parameters have to be experimentally adjusted, to take into account manufacturing uncertainties and unmodeled components.

In the case of the rehabilitation robot presented in [9], the authors focused on offsetting the gravity of the motors. The mass of the rest of the mechanism is neglected compared to the mass of the motors. This assumption is due to the fact that the designed rehabilitation robot for the forearm and the wrist is relatively small. Instead of using the CAD parameters or the known mass of the motors, some authors prefer to measure the torques in a set of positions to estimate the parameters of the gravity model without the need to identify the masses [10]. Note that this approach also uses the static equilibrium equations of the system.

Other methods do not use any equation of the system. For example in [4], the authors define a working area where the compensation of the weight of the exoskeleton is going to be applied and this volume is discretized into small cubes. In each cube, the force is calculated at each vertex so the gravity is compensated and the robot is immobile. Once the force database is completed, and knowing in which cube the exoskeleton is located during the rehabilitation exercise, a weighted mean of the eight vertices of the cube gives the gravity compensation force value.

B. Machine Learning-based Approaches

Machine Learning (ML) is a set of algorithms based on two main ideas: the acquisition of new knowledge from external sources, and the improvement of knowledge representations and structures, so that existing knowledge may be better exploited [11]. In ML, there are many possible techniques and approaches to achieve the same goal, some of them are more appropriate than others for a specific problem. Amongst the existing approaches are: neural networks, Bayesian classifiers, nearest neighbor classifiers, support vector machines and decision trees.

Machine Learning algorithms use computational methods to get information directly from data without relying on a predetermined model. Thus, once a gravity force database is available, ML-based techniques can compute an estimation of the gravity components. In fact, ML has already been used to solve some mechatronic problems such as the design of smart laser welding controllers [12] and adaptive exoskeleton controllers for optimal rehabilitation [13].

III. MATERIALS AND METHODS

Based on the strategies found in the literature, this work analyzes and compares two different solutions to achieve gravity compensation (Fig. 2) and tests them on HomeRehab system. The first one, the analytical method, uses the gravity equations of the device with experimentally fitted parameters. The second solution implements a novel strategy based on Machine Learning to compensate gravity without using the static equations of the device.

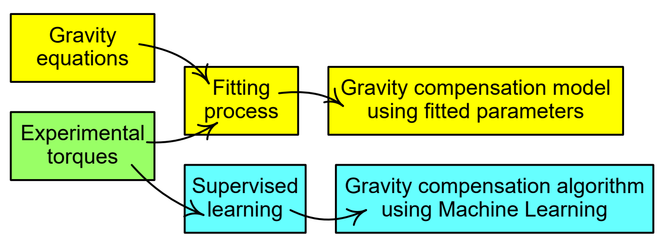

Fig. 2. Gravity compensation strategies

HomeRehab robotic system (Fig. 3) is used to train and test the proposed methods. Table I shows its main technical specifications.

TABLE I
HOMEREHAB SPECIFICATIONS

Workspace	800 $\times$ 400 $\times$ 400 mm
Maximum force	14 N (cont.), 28 N (peak)
Actuators	Maxon DCX32L-GPX32 128 mN·m, 4:1
Transmission	20:1 cable transmission
Encoders	Maxon ENX16, 1024 ppr
Position resolution	6 μm
Weight	8.2 kg

Manuscript 3074 submitted to 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received May 13, 2020.

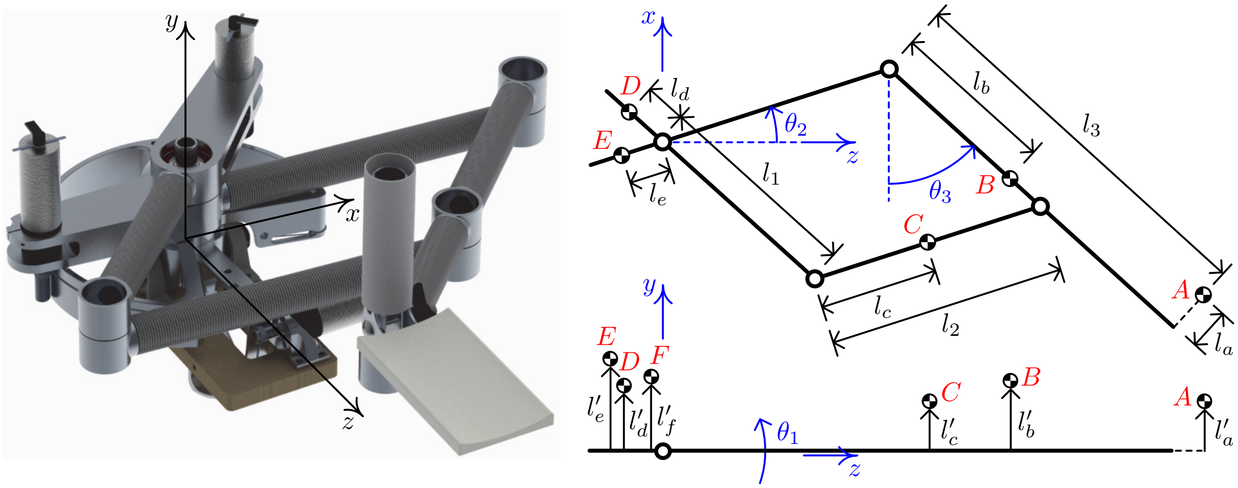

Fig. 3. Scheme of the pantograph and positions of the centers of gravity

HomeRehab has the option to work both in 2D and 3D workspace. When working in 2D, the patient exercises with HomeRehab sitting in a chair and training 2D movements in a planar workspace. Once the patient is able to control and hold the weight of its arm, our device lets him exercise on virtual Daily Life Activities in a 3D workspace, standing in front of the system, and holding its end-effector as a traditional haptic device. In this case, the proposed active gravity compensation methods aim to overcome the weight of the device so the exercises are more realistic and less tired.

To derive the static equilibrium equations of HomeRehab, it is assumed that the centers of gravity of the links are placed along their axis, but not necessarily at their geometric centers (unknown distances l_b, l_c, l_d and l_e), and also at different heights with respect to the plane of rotation of the pantograph (l'_b, l'_c, l'_d and l'_e). The lengths of the mechanism are $l_1 = 0.2$ m, $l_2 = 0.3$ m and $l_3 = 0.4$ m. Angle θ_1 is the rotation of the mechanism with the horizontal plane, while angles θ_2 and θ_3 define the position of the pantograph links.

The handle is modeled as a punctual mass (point A), and the gravity centers of the driving links (points D and E) are located closer to the ends where the motors are anchored. Note that the pulley of the mechanism rotates in solidarity with the pantograph. Its center of gravity (point F) is not over axis x, and it is also outside the pivoting plane of the pantograph.

Using the scheme of Fig. 3 and operating the static equilibrium equations, the gravity components of the device are:

$$\begin{aligned}\tau_1 &= -p_1 \mathrm{s}_1 + p_2 \mathrm{c}_1 \mathrm{c}_2 + p_3 \mathrm{c}_1 \mathrm{s}_3 - p_4 \mathrm{c}_1 + p_5 \mathrm{c}_1 \mathrm{c}_3 \\ \tau_2 &= -p_2 \mathrm{s}_1 \mathrm{s}_2 \\ \tau_3 &= p_3 \mathrm{s}_1 \mathrm{c}_3 - p_5 \mathrm{s}_1 \mathrm{s}_3\end{aligned} \tag{1}$$

where $\mathrm{s}_i = \sin\theta_i$ and $\mathrm{c}_i = \cos\theta_i$. These components depend on five parameters, p_1, p_2 p_3, p_4 and p_5, which in turn depend on the masses and positions of the centers of gravity of the links:

$$\begin{aligned}p_1 &= (m_a l'_a + m_b l'_b + m_c l'_c + m_d l'_d + m_e l'_e + m_f l'_f)\, g \\ p_2 &= (m_a l_2 + m_b l_2 + m_c l_c - m_e l_e)\, g \\ p_3 &= (m_a l_3 + m_b l_b + m_c l_1 - m_d l_d)\, g \\ p_4 &= m_f l_f\, g \\ p_5 &= m_a l_a\, g\end{aligned} \tag{2}$$

A rough estimation of the five parameters p_i could be derived using a CAD model of the mechanism. However, to obtain reliable values for p_i, an experimental fit is required.

ML-based methods do not require the resolution of analytical equations. While HomeRehab is a mechanism whose equations may be derived with relative ease, some parallel mechanisms and commercial devices whose CAD models and geometrical data are not available, the process to develop the equations may be complex.

Among the different ML techniques and approaches, in this work a decision tree technique is used as it allows fast responses and accurate results, as many researchers have already tested [14], [15]. A decision tree is an algorithm and data structure oriented for supervised learning, where each node represents an attribute or feature (in our case, 3D coordinates). For each node, the children are classified according to a criteria until obtaining a leaf node. These leaves will represent the final decision [16].

Usually, several decision trees are used because more accurate results are obtained (each tree may give a different solution and a vote scheme is performed to decide the final decision). The algorithm that achieves this process is called Random Forest [17]. This algorithm can be used to classify or perform a regression prediction, where each tree in the ensemble is trained on a subset of the entire training dataset. Then, each split is performed on a random subset of features (one for each tree) [16].

The Extra Trees are an extension of the random forest regression model proposed by Geurts [18]. Random Forest and Extra Trees are important algorithms within this class and have reported state-of-the-art performance on many

Manuscript 3074 submitted to 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received May 13, 2020.

regression tasks with high-dimensional inputs and outputs [19]. The differences with the original Random Forest are: i) unlike the Random Forest, the Extra Trees does not use the tree bagging step to generate the training subset for each tree, and ii) it randomly selects the best feature along with the corresponding value to split the node [20]. These two differences result in the Extra Trees being less susceptible to overfitting and reporting better performance [18].

It is important to note that the Extra Trees Regression it is not a classification tree. In a regression tree as the target variable does not have classes, we fit a regression model to the target variable using each of the independent variables. Then for each independent variable, the data is split at several points. At each split point, the error between the predicted value and the actual values is squared to get a Sum of Squared Errors (SSE). The split point errors across the variables are compared and the variable/point yielding the lowest SSE is chosen as the root node/split point. This process is recursively continued.

Both methods, analytical and ML-based, need experimental data in order to fit the model or to train the algorithm. The inputs to the methods are the angles measured at the three active joints of the device in any Cartesian position of its workspace, while the outputs are the three torques that have to be applied to each joint to hold the device still in each Cartesian position. For other devices, the number of inputs and outputs should be equal to the number of active degrees of freedom of the device. To generate such force and position database, the workspace of the device is divided into a finite number of points, and an experiment is designed to compute the torque values necessary to hold the system still in each point.

IV. EXPERIMENTS

This section first describes how the force database is collected, and the procedure to fit the analytical equations for gravity compensation and to train the ML-based method. A final experiment is carried out on a set of points of HomeRehab's workspace, different from the points used for the training, to compare the performance of each method.

A. Experimental Setup

The workspace of HomeRehab is divided into 4920 points covering 79.4 % of the workspace used for rehabilitation applications with HomeRehab. The tested positions (Fig. 4) form a rectangular parallelepiped grid in the Cartesian axes, 0.8 m $\times$ 0.22 m $\times$ 0.18 m. The distance between points is 2 cm.

A PID controller forces the system to move automatically from one position to another. When the mechanism reaches the steady state at each tested position (position error below 0.1 mm), the torques provided by the controller are precisely the torques that compensate the gravity. These torques τ_1, τ_2 and τ_3 are recorded in a database, together with the angles θ_1, θ_2 and θ_3 measured at each active joint of the device. The generation of this database takes two hours and a half.

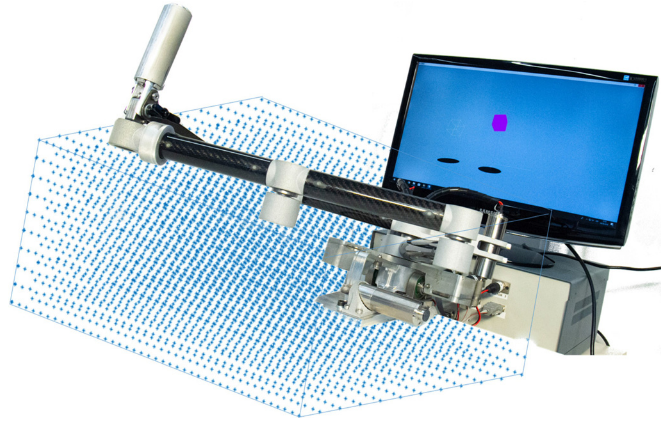

Fig. 4. Tested positions

B. Analytical Model

A least-squares optimization method is carried out to obtain the values of p_i that best fit model (1) with the training database. The following values are obtained:

$$
\begin{aligned}
p_1 &= 3.33 \ \text{N}\cdot\text{m} \\
p_2 &= 3.97 \ \text{N}\cdot\text{m} \\
p_3 &= 3.98 \ \text{N}\cdot\text{m} \\
p_4 &= 2.05 \ \text{N}\cdot\text{m} \\
p_5 &= 0.77 \ \text{N}\cdot\text{m}
\end{aligned} \tag{3}
$$

The experimental torques and the torques that are obtained from the gravity model (1) with the proposed parameters (3) are shown in Fig. 5 for 2000 consecutive points of the training database. It is worth noting that the gravity model experimentally fitted is robust to some assumptions of the scheme depicted in Fig. 3, because the estimated parameters p_i contain the contribution of several distributed masses. It is not especially relevant for the validity of the model to find the exact value of each length and mass of the model.

Taking into account that the DC motors of the mechanism can exert up to 10.24 N·m after the transmission, a non-negligible amount of torque is used to compensate the gravity of the device. In the case of torque τ_1, which is the worst case of the three motors (Fig. 5), the mean value within all the positions of the workspace is 2.27 N·m, which represents the 22 % of the maximum continuous torque. The maximum gravity torque τ_1 is 3.91 N·m, 38 % of the available torque.

C. Machine Learning based method development

The input data for the ML algorithm is the same as for the analytical method (the 4920 training points). This data is arranged in six columns (x, y, z, τ_1, τ_2, τ_3). There was no editing, cleaning or any other technique used on the dataset. The ML method development consisted of evaluating several scenarios for the Extra Tree algorithm, changing the % of data for training and testing. The ML algorithm is implemented in Python 3.6 using the environment Anaconda 5.0.1 x64 and the library Scikit-learn 0.19. Table II shows the results of applying Extra Trees algorithm to different scenarios.

Manuscript 3074 submitted to 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received May 13, 2020.

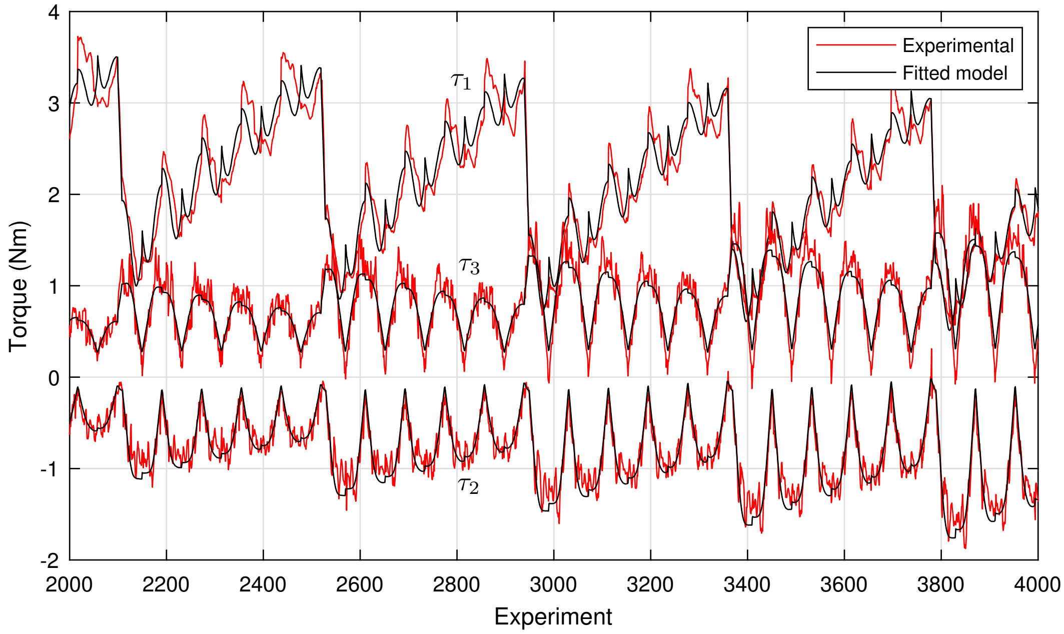

Fig. 5. Experimental and theoretical torques of the fitted model

The tests were performed using a fixed seed for NumPy 12345 in order to be reproducible. Accuracy was evaluated using other different seeds, but there was no significant difference in the results. The base function used was *ExtraTreesRegressor* with 50 estimators. It is relevant to consider that increasing this parameter adds complexity on the calculation adding trees into the forest, and we tried to increase it with no significant improvement, but with high penalty on point calculation. The maximum features are 3, since the dataset only has 3 inputs. Finally, the random state is set to 0.

TABLE II

RESULTS FOR DIFFERENT SCENARIOS OF EXTRA TREE ALGORITHM

Scenario	1	2	3	4
Training	80%	40%	20%	5%
Testing	20%	60%	80%	95%
R^2	0.958	0.943	0.931	0.899

From Table II it can be seen that using the 80% of data for training gets the best results. Other approaches such as Random Forest Regressor, Decision Tree Regressor and others, were also tested with those datasets to check our initial hypothesis of using the Extra Trees Regression method. The results were not improved in terms of accuracy (coefficient of determination R^2 in Table III). However, note that MultiO/P GBR method achieves good performance with lower prediction time and memory usage. Thus, if real-time implementation specifications are very relevant (below 1 ms), this last method would be the preferred choice.

D. Validation

A validation experiment is carried out with HomeRehab to test the performance of both methods. Analytical gravity compensation equations are directly programmed in the NI MyRIO controller of HomeRehab, that runs at a sampling time of 1 ms. A local-host Python server is created as a middleware between the PC that runs the ML algorithm and the controller of HomeRehab. The input message for the server is the actual position of the end-effector (three angles) and the output are the torques needed in the three motors to compensate the gravity force of the mechatronic device. Communication between the computer and the NI MyRIO controller is achieved by UDP protocol, sending end-effector positions from the controller to the computer and receiving compensation torques computed by the ML algorithm. Average computation time for ML prediction and UDP communication is approximately 2-3 ms.

The validation experiment consists of moving HomeRehab system to 120 points (84 points inside the limits of the training cube, 36 points outside the training cube). None of these points were previously used for the training phase. These 120 points result from the combination of $x = [-0.53, -0.43, -0.33, -0.23, -0.13, -0.03, 0.03, 0.13, 0.23, 0.43]$, $y = [0.07, 0.11, 0.15]$, and $z = [0.29, 0.33, 0.37, 0.41]$ (m). 36 of the 120 points are outside the training cube (do not satisfy $-0.4 < x < 0.4$ m).

The validation test is carried out as follows: First, a PID controller holds the device still in the selected point with a position error below 0.1 mm. The torque values computed by the PID to hold the device still in each one of the points are considered the ground truth data for validation. Once the system reaches each point, we replace the torques computed by the PID controller by the torques derived from

TABLE III
RESULTS FOR DIFFERENT ML ALGORITHMS

Name	MSE	R^2	Training time (mean)	Prediction time (mean)	Memory (MB)
Extra Trees	0.0156	0.9489	1.05 s	2.64 ms	278.61
K-Nearest Neighbors	0.0213	0.9280	1.87 ms	353 μs	278.81
Linear Regression	0.0655	0.7516	465 μs	45.3 μs	279.15
Ridge Regression	0.0655	0.7509	1.37 ms	43.9 μs	279.57
Lasso	0.3305	-	598 μs	50.6 μs	279.61
Random Forest Reg.	0.0176	0.9419	667 ms	5.32 ms	279.61
Decision Trees	0.0363	0.8879	9.85 ms	50.1 μs	271.80
Multi Output Gradient Boosting Reg.	0.0199	0.9342	1.39 s	747 μs	271.92

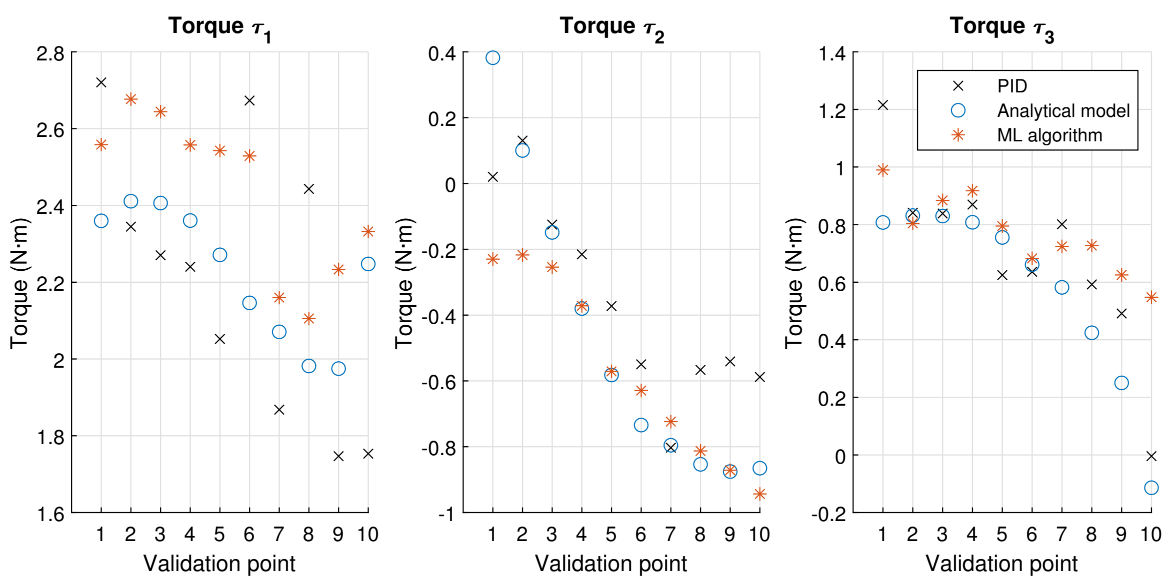

Fig. 6. Torque values computed by the PID controller and both methods (analytical model and Machine Learning algorithm) in a set of 10 validation points.

the proposed two methods, and we evaluate whether the device remains immobile in the same position or if it moves (either falling or moving in other directions).

The outcomes of this experiment show that both methods behave properly holding the device in all the 84 points that lie within the training workspace. However, the ML-based method fails to hold the device still in 25 points outside the training workspace while the analytical method succeeds in all 36 points outside the training workspace.

Fig. 6 shows an example of the torque values computed by both methods along ten validation points: $x = [-0.53, -0.43, -0.33, -0.23, -0.13, -0.03, 0.03, 0.13, 0.23, 0.43]$, $y = 0.11$, and $z = 0.37$ (m). It can be seen from Fig. 6 that torque values for both methods, compared with the PID torque values, are slightly different in each point. However, HomeRehab remains still in most of them, which is considered a good behavior. This fact is due to the friction components of the mechanism that need to be overcome to start moving.

In this particular set of 10 points, the analytical equations are able to hold the system still in all of them. However, the ML-based method fails on points 1 and 10, that are outside the limits of the training cube (point 2 is also outside the limits but the method performs well). At these two points, the robot does not hold still with the ML-based method.

Fig. 7 shows the box plots visualizing the error in all 120 points between the PID values and the values given by both methods. For each joint and method, two set of box plot figures are shown, one for the 84 points inside the limits of the training cube (in black) and the other set for the remaining points outside the limits (in blue). It can be seen that while ML-method computes similar values to the analytical method inside the training cube (except for joint 1), outside the cube joint 2 and 3 torque values are different.

V. DISCUSSION

From the outcomes of the validation experiment it is difficult to compare which one of the two methods provides better gravity compensation torque values. There is a range of

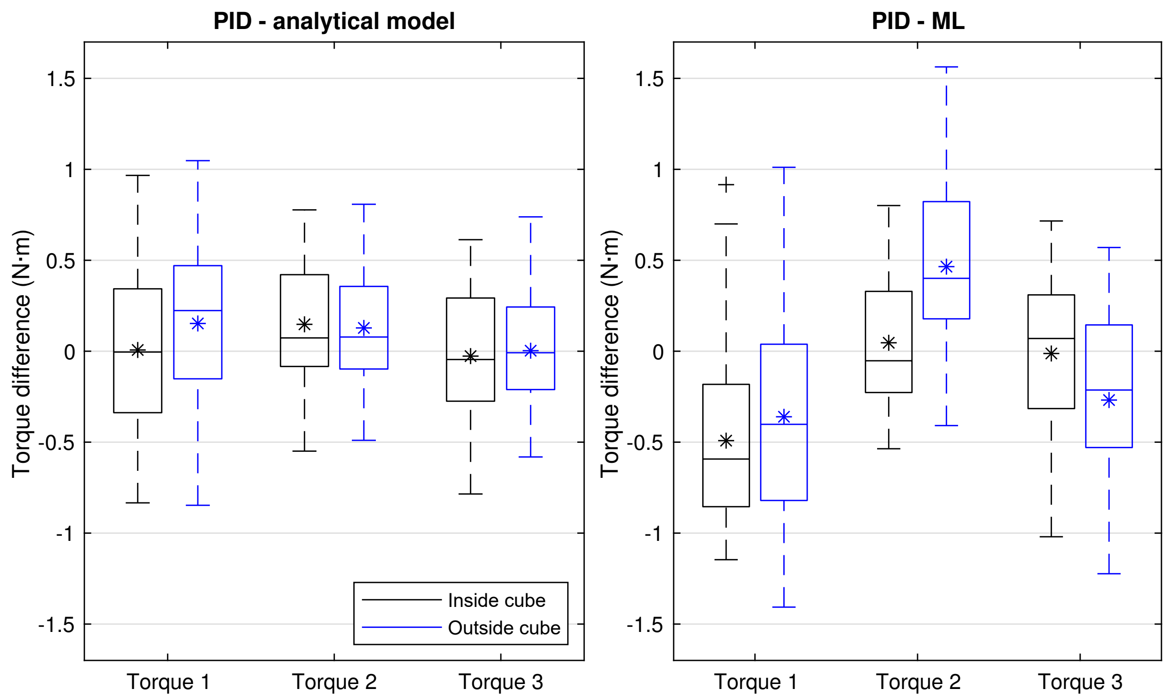

Fig. 7. Box plots of the torque difference between the PID values and the values given by both methods (points within the training cube in black and points outside the training cube in blue) and mean values (asterisks).

torques where the results can be considered valid, that is, the system holds still. This range of torques directly depends on the friction of the system. Even the PID torque values, used as ground truth, are affected by the friction. Nevertheless, the qualitative result of whether the system holds still or not is a valid outcome for the aim of compensating the gravity forces of the device.

Training data is obtained relatively fast, two hours and a half, for a medium-size workspace used for upper-limb rehabilitation. The analytical method seems to behave better as it can give proper results even outside of the workspace covered by the training data. However, deriving the gravity equilibrium equations may not be always possible or easy, e.g. for some parallel mechanisms. ML-based method allows the implementation of an algorithm that overcomes this drawback as it does not require the analytical equations of equilibrium, but it does not perform so well outside the workspace considered in the training data.

The drawback of the ML method may not be relevant if the training data covers the entire workspace of the mechanism. However, this circumstance should be considered for large and complex workspaces. The fact that the ML-method performs poorly outside the area of the training data sounds like overfitting. We performed several experiments with different parameters (estimators, seeds, train test split, etc.) to further analyze the issue, but we discovered that it was not overfitting but lack of precision on the method. Therefore, we think that using Deep Learning methods may lead to better results. The goal of this paper was to focus specifically on traditional ML methods, but future work will evaluate the performance of Deep Learning methods as a solution to the poor performance of the ML method outside the workspace considered on the training data.

A video showing the behavior of the system using the compensation methods is available as a supplement material of this work. The ML-based method code and the training dataset are also available to the public.

VI. CONCLUSION

Gravity compensation is a mandatory feature of mechatronic devices used for rehabilitation. People with limited mobility cannot manipulate bulky apparatus and should be able to perform rehabilitation tasks without extra impediments, as if they were moving their limbs freely.

In this article, we describe the use of Machine Learning methods to ease the complex task of developing proper active gravity compensation control algorithms for robotic rehabilitation. Gravity compensation is of paramount importance to achieve transparent haptic interactions specially for medium and large-size mechatronic devices where the inertia of the system is not negligible in free motion. Traditional control methods for active gravity compensation require to derive the gravity equilibrium equations of the system that may not be straightforward for complex kinematic configurations. This work proposes a gravity compensation strategy based on Machine Learning methods as an easy and fast approach to the problem. It describes its implementation and it compares the method with the traditional approach that it is also described in detail. In general, the results show that the traditional analytical solution performs better than the ML-approach. However, in situations where it is difficult to derive an analytical solution but obtaining training data for ML approaches is considerably easier, results shows that ML-models offer a promising alternative with certain limitations regarding workspace coverage. Results can also be extended to other robotic and haptic domains. Furthermore, we believe that Machine Learning can also be applied to other rehabilitation tasks such as the monitoring of the patients progress by gathering and analyzing multiple robot sensor measures during the exercises, and personalizing the assistance control algorithms for each individual patient and exercise.

References

[1] A. M. Okamura, M. J. Mataric, and H. I. Christensen, "Medical and health-care robotics," *IEEE Robotics Automation Magazine*, vol. 17, no. 3, pp. 26–37, September 2010.

[2] J. C. Pulido, C. Suarez-Mejias, J. C. Gonzalez, A. Duenas Ruiz, P. Ferrand Ferri, M. E. Martinez Sahuquillo, C. E. Ruiz De Vargas, P. Infante-Cossio, C. L. Parra Calderon, and F. Fernandez, "A socially assistive robotic platform for upper-limb rehabilitation: A longitudinal study with pediatric patients," *IEEE Robotics Automation Magazine*, vol. 26, no. 2, pp. 24–39, June 2019.

[3] I. Díaz, J. M. Catalan, F. J. Badesa, X. Justo, L. D. Lledo, A. Ugartemendia, J. J. Gil, J. Díez, and N. García-Aracil, "Development of a robotic device for post-stroke home tele-rehabilitation," *Advances in Mechanical Engineering*, vol. 10, no. 1, pp. 1–8, 2018.

[4] R. Ott, M. Gutierrez, D. Thalmann, and F. Vexo, "Improving user comfort in haptic virtual environments through gravity compensation," in *World Haptics Conference*, 2005, pp. 401–409.

[5] M. Liu and N. H. Quach, "Estimation and compensation of gravity and friction forces for robot arms: Theory and experiments," *Journal of Intelligent and Robotic Systems*, vol. 31, no. 4, pp. 339–354, 2001.

[6] D. Checcacci, E. Sotgiu, A. Frisoli, C. A. Avizzano, and M. Bergamasco, "Gravity compensation algorithms for parallel haptic interface," in *11th IEEE Int. Workshop on Robot and Human Interactive Communication*, 2002, pp. 140–145.

[7] G. Spagnuolo, M. Malosio, A. Scano, M. Caimmi, G. Legnani, and L. M. Tosatti, "Passive and active gravity-compensation of lightarm, an exoskeleton for the upper-limb rehabilitation," in *IEEE Int. Conf. on Rehabilitation Robotics*, 2015, pp. 440–445.

[8] D. G. Chung, M. Hwang, J. Won, and D.-S. Kwon, "Gravity compensation mechanism for roll-pitch rotation of a robotic arm," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016, pp. 338–343.

[9] L. Luo, L. Peng, Z. Hou, and W. Wang, "Design and control of a 3-dof rehabilitation robot for forearm and wrist," in *39th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society*, 2017, pp. 4127–4130.

[10] S. Moubarak, M. T. Pham, R. Moreau, and T. Redarce, "Gravity compensation of an upper extremity exoskeleton," in *Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society*, 2010, pp. 4489–4493.

[11] G. Briscoe and T. Caelli, *A compendium of Machine Learning*. Ablex, 1996, vol. I.

[12] J. Günther, P. M. Pilarski, G. Helfrich, H. Shen, and K. Diepold, "Intelligent laser welding through representation, prediction, and control learning: An architecture with deep neural networks and reinforcement learning," *Mechatronics*, vol. 34, pp. 1–11, 2016.

[13] N. Ajjanaromvat and M. Parnichkun, "Trajectory tracking using online learning LQR with adaptive learning control of a leg-exoskeleton for disorder gait rehabilitation," *Mechatronics*, vol. 51, pp. 85–96, 2018.

[14] M. N. Adnan and M. Z. Islam, "A comprehensive method for attribute space extension for random forest," in *17th Int. Conf. on Computer and Information Technology*, 2014, pp. 25–29.

[15] L. Zhang and P. N. Suganthan, "Random forests with ensemble of feature spaces," *Pattern Recognition*, vol. 47, no. 10, pp. 3429–3437, 2014.

[16] A. Heemann, D. Velinova, and M. Gastegger, "Textklassifikation - Ein Wettbewerb (summer term 2017)," 2017.

[17] M. N. Adnan and M. Z. Islam, "Effects of dynamic subspacing in random forest," in *Int. Conf. on Advanced Data Mining and Applications*, 2017, pp. 303–312.

[18] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.

[19] J. Gall, A. Yao, N. Ravazi, L. Van Gool, and V. Lempitsky, "Hough forests for object detection, tracking, and action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2188–2202, 2011.

[20] V. John, Z. Liu, C. Guo, S. Mita, and K. Kidono, "Real-time lane estimation using deep features and extra trees regression," in *Pacific Rim Symp. on Image and Video Technology*, 2015, pp. 721–733.