Neural Models to Predict Irrigation Needs of a Potato Plantation

Mercedes Yartu[1], Carlos Cambra[2], Milagros Navarro[1], Carlos Rad[1], Ángel Arroyo[2], and Álvaro Herrero[2](✉)

[1] Composting Research Group (UBUCOMP), EPS-La Milanera, Universidad de Burgos, C/Villadiego s/n, 09001 Burgos, Spain
{minago,crad}@ubu.es

[2] Departamento de Ingeniería Informática, Grupo de Inteligencia Computacional Aplicada (GICAP), Escuela Politécnica Superior, Universidad de Burgos, Av. Cantabria s/n, 09006 Burgos, Spain
{ccbaseca,aarroyop,ahcosio}@ubu.es

Abstract. Reducing water consumption is an important target required for a sustainable farming. In order to do that, the actual water needs of different crops must be known and irrigation scheduling must be adjusted to satisfy them. This is a complex task as the phenology of plants and its water demand vary with soil properties and weather conditions. To address such problem, present paper proposes the application of time-series neural networks in order to predict the soil water content in a potato field crop, in which a soil humidity probe was installed. More precisely, Non-linear Input-Output, Non-linear Autoregressive and Non-linear Autoregressive with Exogenous Input models are applied. They are benchmarked, together with different interpolation methods in order to find the best combination for accurately predicting water needs. Promising results have been obtained, supporting the proposed models and their viability when predicting the real humidity level in the soil.

Keywords: Time series forecast · Neural networks · Autoregressive · Irrigation · Potato crop

1 Introduction

Originated and first domesticated in the Andes mountains of South America, the potatoes (*Solanum tuberosum*) belongs to the solanaceae family of flowering plants. In terms of agricultural production, potato crop is the third most important food crop in the world after rice and wheat. The EU produced 51.8 million tonnes of potatoes in 2018, with Germany, France, Poland and Netherlands as main producers [1]. In Spain, potato production reaches 2.24 million tonnes, and 40.3% of it is located in Castilla y León, mainly in Burgos (4%) occupying in 2017 around 2,400 ha of irrigated land [2]. In the Mediterranean context, irrigation supposes an extraordinary demand for available water, which constitutes an important problem in a context of water scarcity and climatic

Á. Herrero et al. (Eds.): SOCO 2020, AISC 1268, pp. 600–613, 2021.
https://doi.org/10.1007/978-3-030-57802-2_58

change. The application of innovative and an appropriate transfer of technologies to an adequate management of irrigation is a key factor to reach a sustainable crop production [3].

Monitoring weather variables and water status in soils are key factors to reach minimum water consumption without compromising crop production. The use of satellite or unmanned aerial vehicles (UAVs) imaginaries, automated weather stations and humidity or water potential probes are important tools to achieve precision irrigation adapted to crop phenology [4], which maximizes production avoiding water stresses, lixiviation of nutrients, or the incidence of crop pest and diseases.

In keeping with this idea, present paper proposes the application of IoT and Softcomputing to monitor a potato field crop, located in Cabia (Burgos, Spain), 42°16’57” N and 3°51’25” W, with sprinkler irrigation, to optimize water use efficiency. More precisely, a meteorological station, together with different sensors were placed in the crop in order to gather data in real time. Additionally, some measurements regarding crop development were taken and are analysed in present work. As there is no sensor to measure such features, these measurements must be taken manually. Thus, imaginary figures are not available on a daily basis and hence they must be interpolated in order to merge such data with those gathered through IoT. By taking into account all these data, neural networks for time series forecast are applied in order to predict water needs of the crop.

The remaining sections of this study will be structured as follows: previous work will be discussed in Sect. 2 while the methods applied in present study will be described in Sect. 3. Section 4 will introduce the real-life problem that is addressed, while the obtained results will be presented in Sect. 5. Finally, both conclusions and future work proposals will be discussed in Sect. 6.

2 Previous Work

Artificial Intelligence (AI) in general and Softcomputing in particular, have been previously applied to optimize irrigation systems. As stated in [5], different AI approaches and methods have been studied for smart controlling irrigation systems. More precisely, Neural Networks, Genetic Algorithms, and Fuzzy Logic could lead to optimum utilization of irrigation water resources.

Labbé et al. [6] modelled an irrigation decision process for limited water allocation, a very common pattern and challenge caused by climate change [7], and irrigation scheduling for corn plantations. The model consisted of irrigation management rules for different irrigation-related tasks that were derived from farmer surveys and based on the monitoring of their irrigation practices over a 2-year period. This model was incorporated into a simulator engine that, given the context of the decision, was able to predict irrigation schedules and irrigation volumes with an average error ranging from 6 to 13 mm for different farmers, reflecting an error below 6.7%. Instead of developing a model that captures the farmer’s decision individually, using surveys and observations, in this study the Deep Learning and Artificial Intelligence AI were used to capture the agronomist’s decision process in irrigation system [8].

Meanwhile, authors in [9] proposed a daily irrigation water demand calculation based on an Adaptive Neuro Fuzzy Inference System (ANFIS). This first-order Sugeno fuzzy model is combined with a backpropagation algorithm. It has a better performance (Root Mean Squared Error and Mean Absolute Percentage Error) predicting irrigation needs when compared to the Auto Regressive Moving Average models.

Khan et al. [10] compared different AI models and their error rates when it comes to irrigation prediction. It was found that among all the models, the 3-fold cross validation multiple decision trees SysFor model gave the best overall results. However, the actual amount of water required by the crop was accurately predicted by neural models. The difference in error percentage between ANNs and SysFor was almost 20%. The comparison concluded that SysFor, ANNs, and decision tree techniques are the most suitable ones for the task of irrigation prediction.

A neural network has been applied in [11] to model the temporal (surface) soil moisture fluxes. Only meteorological data and the soil moisture humidity itself are used as input data and no information from the crop is used. Present paper comprises a comprehensive study of different neural networks trained with different algorithms. Furthermore, it is the humidity level in the underground what is precisely predicted so that water demands could be more precisely calculated.

Similarly, non-linear time-series neural networks have been previously applied to some different problems ranging from workplace accidents [12] to road transportation [13] and fault detection [14]. Differentiating from previous works, present paper proposes time-series neural networks in order to predict the humidity in the underground.

3 Applied Methods

As previously stated, two kinds of methods have been applied in present paper; on the one hand, interpolation (described in Subsect. 3.1) has been applied to predict daily values of some features. On the other hand, neural networks (described in Subsect. 3.2) have been applied to predict the humidity level.

3.1 Interpolation

It is widely known that interpolation consists on generating new data points between a given range of values. In order to do that, several alternatives exist for one-dimensional problems. The following ones have been applied in present study:

- Cubic: this is a shape-preserving method for cubic interpolation. Based on the shape of the known data, new values are interpolated by piecewise cubic interpolation, taking into account the values at neighboring grid points.
- Spline: each new value calculated by this method is based on a cubic interpolation of the values at neighboring data in each respective dimension. The not-a-knot end conditions are applied.
- Makima: this modified version of the Akima cubic Hermite interpolation method [15]. Each new value calculated by this method is based on a piecewise function of polynomials (with degree smaller than or equal to 3). In the Akima formula, the value of the derivative at a given data point is a weighted average of nearby slopes. The weights are defined as:

$$w_1 = |\delta_{i+1} - \delta_i|; w_2 = |\delta_{i-1} - \delta_{i-2}| \tag{1}$$

Being δ_i the slope on the interval $[x_i x_{i+1})$. In the modified version, definition of weights is slightly different, as follows:

$$w_1 = |\delta_{i+1} - \delta_i| + \frac{|\delta_{i+1} + \delta_i|}{2}; w_2 = |\delta_{i-1} - \delta_{i-2}| + \frac{|\delta_{i-1} + \delta_{i-2}|}{2} \tag{2}$$

Thanks to that, when two flat regions with different slopes meet, more importance is given to the side where the slope is closer to zero (horizontal), thus avoiding overshoot.

3.2 Neural Models

In order to predict the humidity level, once all data are available (i.e. after interpolation is carried out), 3 neural models for non-linear time-series forecast [16] have been applied, namely: Non-linear Input-Output (NIO), Non-linear Autoregressive (NAR) and Non-linear Autoregressive with Exogenous Input (NARX). These can be seen as feedforward networks in which the input weight has a tap delay line associated with it. Thanks to that, the network has a finite dynamic response to time series input data. The main differences between these 3 models are what data is given to the model in order to predict future values of humidity level. In the case of NIO, it is only the humidity level itself. In the case of NAR, all the other features (described in Sect. 4) except the humidity level are included. Finally, in the case of NARX, these two data sources are considered in the prediction. As a result, the NARX could be mathematically formulated as:

$$y(t) = f\left(y(t-1), \ldots, y\left(t - n_y\right), x(t-1), \ldots, x(t - n_x)\right) \tag{3}$$

Being *y(t)* the variable to be predicted in time instant t, $f()$ the function to be approximated by the neural model, *x(t)* an exogenous variable, n_{y} the maximum number of time delays in the output, and n_{x} the maximum number of time delays in the input. Consequently, the mathematical formulation for the NAR model is:

$$Y(t) = f(y(t-1), \ldots, y(t - ny)) \tag{4}$$

As it can be seen, in the case of the NAR model, the exogenous input (x) is not included in the formulation. Differentiating from this model, the predicted variable is replaced by this exogenous one in the NIO formulation:

$$y(t) = f(x(t-1), \ldots, x(t - n_x)) \tag{5}$$

4 Agronomic Setup

Field experiments were conducted from April 16th to October 10th 2019, in a potato field crop of 5 ha, located in Cabia (Burgos), 42°16’57” N and 3°51’25” W, with a semi-permanent sprinkler irrigation system. Soil was classified as *Calcic Luvisol*

(*LVk*) according to FAO, with loam texture, bulk density 1.26 kg L^{-1}, field capacity 0.31 (w/w), pH (1:5 w/v) 7.6, Electrical Conductivity (1:5 w/v, 25 °C) 0.65 dS m^{-1}, Organic Mater 3.33%, Total N 0.16% and lime 16.7%. Climate in this area is Attenuated Mesomediterranean, according to FAO.

As shown in Fig. 1, an agronomic IoT system was installed in the field, comprising an automatic weather station ATMOS 41 (METTER Group, USA) oriented to North. A soil humidity probe TEROS 10 (METTER) was installed at 15 cm depth, a soil water potential probe TEROS 21 at 30 cm depth and a rain gauge (ECRN 100) were connected to a EM60G data logger, remotely connected with ZENTRA Cloud System (METER Group, USA) that registered data each 30 min.

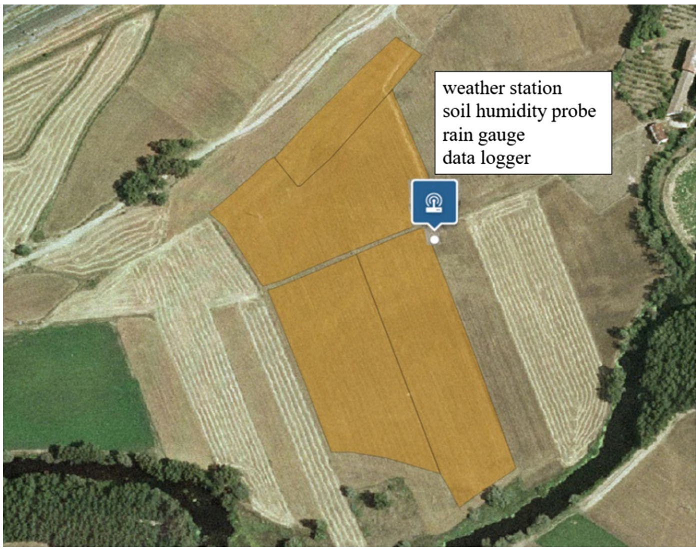


Fig. 1. Field map of the agronomic IoT system.

Potatoes (*Solanum tuberosum* L. Var. Agria) were planted in April 16th and from mid-June, phenological development was assessed according to BBCH-scale and four plants from the centre of the plot (20 × 20 m) were removed for laboratory analysis every 15 days. Morphological parameters such as length of aerial plant, number of stems and leaves, length of roots, number and weight of tubers, wet and dry biomass, chlorophyll content with SPAD, and N-content by a combustion autoanalyzer (TruSpec, LECO) were determined. Before harvesting, four sampling locations of 3 m^2 were chosen at random for yield estimation; tubers were classified by considering their diameter in different commercial classes: >80 mm, between 40–80 mm and <40 mm.

Public imaginary was captured from the satellite SENTINEL-2B under the scope of the EU Copernicus program. Nine images were obtained corresponding to day 11th to 171st, after plant emergency. From them, Normalized Difference Vegetation Index

(NDVI) was calculated according to the equation:

$$NDVI = \frac{(NIR - Red)}{(NIR + Red)} \tag{6}$$

Where Red and NIR are the spectral reflectance measurements acquired in the red (visible) and near-infrared regions, respectively. These data correspond to 4 and 8 of SENTINEL-2B bands, respectively. Raster layers were processed using the software QGIS v. 2.18 to obtain an NVDI vector layer. NVDI data were thereafter transformed into basal crop coefficients (K_{cb}) using equation:

$$K_{cb} = 1.44 \times NDVI - 0.1 \tag{7}$$

Crop evapotranspiration was calculated according to FAO Method 56 approach:

$$ET_C = (K_{cb} + K_s) \times ET_0 \tag{8}$$

Where K_s estimates soil evaporation, which is considered cero during the irrigation period as the crop development quickly cover soil surface.

As a result, the following features are available to apply the neural networks:

- Temperature: gathered from the temperature sensor (−40–50 °C) in the ATMOS 41 Weather Station (Meter Group, USA), Accuracy +/−0.5 °C.
- Precipitation: gathered from the precipitation sensor (0−400 mm/h) in the ATMOS 41 Weather Station (Meter Group, USA), Accuracy +/−5%. Daily
- CCM (Chlorophyl Content Index)[1]: CCM-200 plus Chlorophyll Content Meter (Opti-Sciences, UK) measures optical absorbance in two different wavelengths: 653 nm (Chlorophyll) & 931 nm (Near Infra-Red).
- Plant height[1]: a Carpenters meter (+/−1 mm) was used.
- Plant weight[1]: a weight scale (+/−1 mg) was used.
- % Humidity[1]: weight losses after 38 h at 70 °C (+/−1 °C).
- Aerial part length[1]: a Ruler lab (+/−1 mm) was used.
- Roots length[1]: a Ruler lab (+/−1 mm) was used.
- Plant Nitrogen content[1]: aerial part of plants was dried at 70 °C and thereafter, ground in a mill. Samples of 0.2 g were analysed by Dumas method in a TruSpec CN (LECO, USA) with IRD (Infra-Red Detector) and TCD (Thermal Conductivity Detector) for CO2 and N2, respectively.
- Tubers weight per plant[1]: a weight scale (+/−1 mg) was used.
- Number of tubers per plant[1]: tubers were visually counted.
- Tubers humidity[1]: weight losses after 38 h at 70 °C (+/−1 °C).
- Percentage of tubers in the 0–40 cm diameter range[1]: a squared measurement frame of 40 cm was used.
- Percentage of tubers in the 40–80 cm diameter range[1]: squared measurement frames of 40 and 80 cm were used.

[1] Interpolated by means of the methods described in Subsect. 3.1. All features are interpolated by means of same method each time.

- Percentage of tubers in the >80 cm diameter range[1]: a squared measurement frame of 80 cm was used.
- Tubers Nitrogen content[1]: crushed fresh tubers were dried at 70 °C and thereafter, ground in a mill. Samples of 0.2 g were taken.
- Underground humidity level: Teros 10 (Meter Group). It is a capacitance sensor that determines the dielectric permittivity of soil by measuring the charge time of a capacitor, which uses that medium as a dielectric. The sensor measures the time to charge a capacitor from a starting voltage, V_i to a voltage V_f with an applied voltage, V_f. Its working frequency (70-MHz) minimizes salinity and textural effects in the soil. This is the data feature to be forecast in the range [0, 1].

5 Experiments and Results

The results obtained through the different experiments are described in subsequent subsections. These results are presented by the applied interpolation method (Cubic, Makima, and Spline) and all the applied neural models (NAR, NIO, and NARX). During the experimental study, each one of these models has been tuned with different values of the appropriate parameters:

- Number of input delays: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
- Number of output delays: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
- Number of hidden neurons: {1, 5, 10, 15, 20}
- Training algorithm: {1 - Levenberg-Marquardt, 2 - Batch Gradient Descent, 3 - Gradient Descent with Momentum, 4 - Adaptive Learning Rate Backpropagation, 5 - Gradient Descent with Mo-mentum and Adaptive Learning Rate, 6 - Scaled Conjugate Gradient, 7 - Broyden–Fletcher–Goldfarb–Shanno Backpropagation}

As a result, 350 runs have been performed for the NIO and NAR models, and 3.500 for the NARX model. For each one of them, 10 executions have been carried out in order to obtain more statistically significant conclusions. Average Mean Squared Error (MSE) is provided in each case, calculated as the average MSE of all the included runs and executions. In each one of the tables, the lowest error value per column is in bold.

5.1 Results by Cubic Interpolation

Results (MSE) obtained when applying Cubic interpolation for the given features (listed in Sect. 4) are presented in this section. Firstly, results obtained by the neural models (NAR, NIO, and NARX) are presented per the number of input delays in Table 1.

Similarly, results obtained by the neural models (NAR, NIO, and NARX) are presented per the number of hidden neurons in Table 2.

Table 1. MSE of the results obtained by NAR, NIO, and NARX neural models after Cubic interpolation, averaged results are shown per the number of input delays.

N input delays	NAR	NIO	NARX
1	0.0014689	0.0009734	248.353764
2	0.0010200	0.0008537	293.284653
3	0.0012625	0.0007557	245.431746
4	0.0011707	0.0007253	2114.027857
5	0.0010601	0.0006607	250.708950
6	0.0010487	0.0005999	257.020836
7	0.0011634	0.0006057	513.295907
8	**0.0009762**	0.0005362	**28.8959075**
9	0.0010442	0.0005458	298.689252
10	0.0010452	**0.0005014**	283.937783

Table 2. MSE of the results obtained by NAR, NIO, and NARX neural models after Cubic interpolation, averaged results are shown per the number of hidden neurons.

N neurons	NAR	NIO	NARX
1	**0.00046848**	0.00050660	1096.76566
5	0.00085599	0.00170161	**188.035696**
10	0.00106059	0.00053109	282.978015
15	0.00149436	0.00032107	421.410713
20	0.00175055	**0.00031854**	406.349854

Finally, results obtained by the neural models (NAR, NIO, and NARX) are presented per the training algorithm in Table 3.

From the results obtained by Cubic interpolation, it can be said that NARX obtained, by far, the worst results in terms of error (MSE). When considering the number of input delays, the lowest error was obtained by the NIO model, with the highest number of delays (10). The lowest error for each one of the other neural models was also obtained with a high number of delays (8). After comparing the obtained results per number of hidden neurons, it is worth mentioning the best results in terms of MSE are obtained by NIO model comprising 20 neurons in the hidden layer. Finally, the training algorithm that outperforms all the other ones for the three neural models is Levenberg-Marquardt (LM). The lowest error when applying this algorithm is obtained by the NAR model.

Table 3. MSE of the results obtained by NAR, NIO, and NARX neural models after Cubic interpolation, averaged results are shown per the training algorithm.

Training algorithm	NAR	NIO	NARX
1	**0.000160087**	**0.000175692**	**8.623729042**
2	0.003173096	0.001686197	2215.944487
3	0.003385888	0.001619598	1016.390437
4	0.000293968	0.000286203	30.48441364
5	0.000353738	0.000360591	56.31164187
6	0.000202528	0.000249584	9.393541776
7	0.000312652	0.000352601	16.60766226

5.2 Results by Makima Interpolation

In a way like previous subsection, results (MSE) obtained when applying Makima interpolation are presented in this section. Firstly, results obtained by the neural models (NAR, NIO, and NARX) are presented per the number of input delays in Table 4.

Table 4. MSE of the results obtained by NAR, NIO, and NARX neural models after Makima interpolation, averaged results are shown per the number of input delays.

N input delays	NAR	NIO	NARX
1	**0.00016344**	0.00069287	0.00089254
2	0.00111653	0.00031333	0.00081513
3	0.00385679	0.00101315	0.00073656
4	0.00303471	0.00029949	0.00072213
5	0.00164207	0.00110668	0.00065707
6	0.00033552	**0.00029739**	0.00060338
7	0.00037262	0.00118705	0.00058923
8	0.00022522	0.00031471	0.00055197
9	0.00023622	0.00124011	0.00054781
10	0.00029200	0.00034244	**0.00053628**

Table 5 shows results obtained by the neural models (NAR, NIO, and NARX) presented per the number of neurons in the hidden layer of the models.

Table 5. MSE of the results obtained by NAR, NIO, and NARX neural models after Makima interpolation, averaged results are shown per the number of hidden neurons.

N neurons	NAR	NIO	NARX
1	**0.000463906**	**0.00050310**	**0.000488115**
5	0.000848853	0.00065632	0.000636506
10	0.001128539	0.00070203	0.000690203
15	0.001465014	0.00075088	0.000748559
20	0.001731256	0.00079128	0.000762679

Finally, results obtained by the neural models (NAR, NIO, and NARX) after Makima interpolation are presented per the training algorithm in Table 6.

Table 6. MSE of the results obtained by NAR, NIO, and NARX neural models after Makima interpolation, averaged results are shown per the training algorithm.

Training algorithm	NAR	NIO	NARX
1	**0.00016150**	**0.000180876**	**0.00014807**
2	0.00332818	0.000802008	0.00172820
3	0.00325209	0.000274582	0.00166450
4	0.00030008	0.000878147	0.00025697
5	0.00075088	0.000928166	0.00033010
6	0.00021049	0.000605822	0.00020037
7	0.00028627	0.000695771	0.00032827

After analyzing results in Tables 4, 5 and 6, it is worth mentioning that NAR and NARX models obtained the best results. When considering the number input delays, the minimum value (1) lead the NAR model to obtain the lowest error. In the case of NIO and NARX, 6 and 10 input delays respectively caused the models to reduce the error to the minimum. Regarding the number of hidden neurons, results are very consistent as the three models obtained the lowest MSE value when comprising only one hidden neuron. As it has been highlighted in the case of Cubic interpolation, LM is the training algorithm that let the models to obtain the minimum error when applied to Makima-interpolated data.

5.3 Results by Spline Interpolation

Finally, results (MSE) obtained when applying Spline interpolation are presented in this section. Firstly, Table 7 shows results obtained by the neural models (NAR, NIO, and NARX), presented per the number of input delays.

Table 7. MSE of the results obtained by NAR, NIO, and NARX neural models after Spline interpolation, averaged results are shown per the number of input delays.

N input delays	NAR	NIO	NARX
1	**0.00016068**	0.00098305	0.000937062
2	0.00095830	0.00083037	0.000805622
3	0.00394303	0.00083393	0.000741996
4	0.00274118	0.00067616	0.000675199
5	0.00167735	0.00064080	0.000646854
6	0.00031176	0.00062095	0.000613788
7	0.00035929	0.00060266	0.000595541
8	0.00029289	0.00057446	0.000548307
9	0.00024224	0.00054343	**0.000238717**
10	0.00031556	**0.00054079**	0.000538653

Similarly, results obtained by the neural models (NAR, NIO, and NARX) are presented per the number of hidden neurons in Table 8.

Table 8. MSE of the results obtained by NAR, NIO, and NARX neural models after Spline interpolation, averaged results are shown per the number of hidden neurons.

N neurons	NAR	NIO	NARX
1	**0.00044984**	**0.000512456**	**0.0004855**
5	0.00076514	0.000647422	0.0006236
10	0.00110439	0.000695201	0.0007031
15	0.00143401	0.000809941	0.0007577
20	0.00174776	0.000775962	0.0007533

Table 9 shows the results obtained by the neural models (NAR, NIO, and NARX), presented per the training algorithm.

It is worth mentioning that from the results obtained by Spline interpolation, as it happened in the case of Makima interpolation, best results have been obtained by NAR and NARX. Only one input delay was used by NAR to get the lowest error rate, while NIO and NARX employed high values (ten and nine respectively). As it happened when analyzing Makima-interpolated data, the three models obtained the lowest error when configured with one hidden neuron. NAR and NARX obtained the lowest error when they have been trained with the LM algorithm. Differentiating from these models, the lowest error was obtained by NIO on Spline-interpolated data when been trained with the Scaled Conjugate Gradient algorithm.

Table 9. MSE of the results obtained by NAR, NIO, and NARX neural models after Spline interpolation, averaged results are shown per the training algorithm.

Training algorithm	NAR	NIO	NARX
1	**0.00015990**	0.000947065	**0.0001458**
2	0.00328746	0.001717524	0.0017001
3	0.00305368	0.001668056	0.0016993
4	0.00029515	0.000302221	0.0002572
5	0.00039139	0.000350443	0.0003258
6	0.00020699	**0.000246865**	0.0001987
7	0.00030703	0.000355659	0.0003256

Finally, as in the case of the NARX model, both input and output delays are applied, Table 10 presents the results obtained by this model per interpolation method and number of output delays.

Table 10. MSE of the results obtained by NARX neural model, averaged results are shown per the number of output delays and interpolation method.

N output delays	Cubic	Makima	Spline
1	**253.2822868**	0.000678297	0.000677078
2	268.6515364	0.000667616	0.000670093
3	393.6519372	0.000653401	0.000674038
4	2120.995545	0.000682745	**0.000160276**
5	350.7752469	0.000667075	0.000669579
6	273.7858462	**0.000633245**	0.000666098
7	273.7598735	0.000656846	0.000656566
8	257.0172385	0.000666048	0.000658993
9	279.7134575	0.000684088	0.000666557
10	319.4469076	0.000662762	0.000667852

In this table it can be seen, as previously mentioned, that NARX obtained very bad results (high error rates) when applied to Cubic interpolation. On the contrary, acceptable results were obtained with medium values of output delays (6 and 4) in the case of Makima and Spline interpolation respectively.

6 Conclusions and Future Work

In general terms it can be said that present research has successfully addressed the initial targets. Different interpolation methods and time-series neural models have been combined and benchmarked in order to accurately predict the soil humidity level in a potato field. Thanks to the experimental validation, irrigation needs of the studied plantation could be adjusted.

After analyzing the presented results, conclusions can be derived from both the softcomputing and agricultural perspective. Taking into account the applied methods, it can be said that:

- The interpolation methods do not have a significant effect on the prediction except in one case. The NARX model when applied to Cubic-interpolated data obtained very high error rates.
- There is not a neural model that clearly outperforms the other ones; NIO obtained most best results when applied to Cubic-interpolated data while NAR and NARX outperformed it when applied to Makima and Spline-interpolated data. Furthermore, the parameter tuning of each model must be adjusted to each case as there is not a given combination of parameters that always leads to best results. The clearest conclusion about parameter tuning is that Levenberg-Marquardt is the best option when selecting the training algorithm. Except in one case (NIO applied to the Spline-interpolated data), it led the models to get the lowest error rates.

Actually, the activity of country-based institutional services involved in helping farmers to manage irrigation practices, are based only in forecast predictions, being more difficult to introduce in them predictions based in the available water content of the soil. The installation of non-expensive soil stations, with soil humidity probes located in reference soil profiles and covering wide irrigation areas, and the use of time series methods and neural networks for data analysis, would considerately improve soil water content monitoring and irrigation predictions.

As a proposal for future work, authors suggest applying some other softcomputing models to improve forecast. On the other hand, more input features may be considered and feature selection methods would be applied in order to identify those ones that are more important in order to predict the level of underground humidity.

Acknowledgements. : This work was financed by a grant agreement between Lab-Ferrer and UBUCOMP. Authors are grateful to the farmer Mr. José María Izquierdo for providing the experimental field and the monitoring of irrigation.

References

1. Agricultural Production Crops. https://ec.europa.eu/eurostat/statistics-explained/index.php/Agricultural_production_-_crops#Potatoes_and_sugar_beet. Accessed 02 Sept 2020
2. Yearly Statistics. https://www.mapa.gob.es/es/estadistica/temas/publicaciones/anuario-de-estadistica/2018/default.aspx?parte=3&capitulo=07&grupo=3&seccion=2. Accessed 02 Sept 2020

3. Pereira, L.S., Oweis, T., Zairi, A.: Irrigation management under water scarcity. Agric. Water Manag. **57**, 175–206 (2002)
4. Althoff, D., Alvino, F.C.G., Filgueiras, R., Aleman, C.C., da Cunha, F.F.: Evapotranspiration for irrigated agriculture using orbital satellites. Bioscience Journal **35**, 670–678 (2019)
5. Shitu, A., Tadda, M., Danhassan, A.: Irrigation water management using smart control systems: a review. Bayero Journal of Engineering and Technology **13**, 2449–2539 (2018)
6. Labbé, F., Ruelle, P., Garin, P., Leroy, P.: Modelling irrigation scheduling to analyse water management at farm level, during water shortages. Eur. J. Agron. **12**, 55–67 (2000)
7. Fry, A.: Water: facts and trends. World Business Council for Sustainable Development (2006)
8. Andriyas, S., McKee, M.: Recursive partitioning techniques for modeling irrigation behavior. Environ. Model Softw. **47**, 207–217 (2013)
9. Atsalakis, G., Minoudaki, C., Markatos, N., Stamou, A., Beltrao, J., Panagopoulos, T.: Daily irrigation water demand prediction using adaptive neuro-fuzzy inferences systems (anfis). In: Proceedings 3rd IASME/WSEAS International Conference on Energy, Environment, Ecosystems and Sustainable Development, pp. 369–374. WSEAS (2007)
10. Khan, M.A., Islam, M.Z., Hafeez, M.: Evaluating the performance of several data mining methods for predicting irrigation water requirement. In: AusDM, pp. 199–208 (2012)
11. Adeyemi, O., Grove, I., Peets, S., Domun, Y., Norton, T.: Dynamic neural network modelling of soil moisture content for predictive irrigation scheduling. Sensors **18**, 3408 (2018)
12. Contreras, S., Manzanedo, M.Á., Herrero, Á.: A hybrid neural system to study the interplay between economic crisis and workplace accidents in Spain. Journal of Universal Computer Science **25**, 667–682 (2019)
13. Alonso de Armiño, C., Manzanedo, M.Á., Herrero, Á.: Analysing the intermeshed patterns of road transportation and macroeconomic indicators through neural and clustering techniques. Pattern Anal. Appl. **23**(3), 1059–1070 (2020). https://doi.org/10.1007/s10044-020-00872-x
14. Taqvi, S.A., Tufa, L.D., Zabiri, H., Maulud, A.S., Uddin, F.: Fault detection in distillation column using NARX neural network. Neural Comput. Appl. **32**(8), 3503–3519 (2018)
15. Akima, H.: A method of bivariate interpolation and smooth surface fitting for irregularly distributed data points. ACM Trans. Math. Softw. **4**, 148–159 (1978)
16. Leontaritis, I.J., Billings, S.A.: Input-output parametric models for non-linear systems Part I: deterministic non-linear systems. Int. J. Control **41**, 303–328 (1985)

Special Session: Soft Computing Applied to Robotics and Autonomous Vehicles

Mathematical Modelling for Performance Evaluation Using Velocity Control for Semi-autonomous Vehicle

Khayyam Masood(✉), Matteo Zoppi, and Rezia Molfino

PMAR Robotics, DIME, University of Genova, Genova, Italy
khayyam.masood@edu.unige.it

Abstract. Freight Urban RoBOTic vehicle (FURBOT) is a semi autonomous vehicle for which it is desired that it could deliver freight autonomously from one destination to another. The vehicle is required to operate in Genova, Italy which in general has steep slopes. Additionally, safety of this vehicle and of the environment is of critical importance for urban autonomous driving thus the need for having a simulation model arises. Furthermore, the vehicle is expected to perform last mile freight delivery in European H2020 project SHOW for which highest autonomy is required. For these purposes, a mathematical model is constructed for autonomous velocity control over gradient varying hilly terrain. Autonomous traction and braking of the vehicle is introduced for catering for gradient varying terrain. The model built for this vehicle will serve as basis for embedding new sensors in future, tracking their performance and overall creating a safe environment for the vehicle to operate.

Keywords: Mathematical modeling · Performance evaluation · Mobile robots · Freight vehicle · Velocity control

1 Introduction

Freight Urban RoBOTic vehicle (FURBOT) is light weight, fully electronic vehicle designed for sustainable freight transport in urban areas. It is one of pioneering autonomous vehicles in freight delivery sector. The vehicle is expected to handle first and last mile freight delivery in an urban environment setting for the European H2020 project SHOW (SHared automation Operating models for Worldwide adoption). For the project SHOW, FURBOT is expected to attain maximum autonomy in its drive. Due to the autonomy requirements of SHOW project, it is essential for FURBOT to be modelled and simulated prior at length. For this purpose, it is very essential to build a custom-made simulation platform where automation testing and vehicle performance could be judged prior to experiments. This work is an effort to create such simulation platform in order to enhance the performance of the vehicle when integrated with new sensors and in general observing the performance anomalies if any.

Á. Herrero et al. (Eds.): SOCO 2020, AISC 1268, pp. 617–626, 2021.
https://doi.org/10.1007/978-3-030-57802-2_59

Dynamic modelling of new types of robots is becoming an essential research platform and is a required essential tool for developing better performing robots in current research. From Bi-pedal robots [1] to Hexa-slide robots [2] to parallel manipulators [3], dynamic modeling is becoming the key for better understanding the robots behavior in the natural environment.

Motion dynamics and custom control in robotics is essential for performance evaluation. Mobile robots specially wheeled robots which have vast human interaction and unknown environment requires simulation testing. Reference [4] uses sliding mode control for trajectory tracking for wheeled robot and [5] uses control to move their robot on slippery downhill. Mathematical modeling thus mainly revolves around modeling Newton-Euler equations [6,7] and using MATLAB as a tool for simulation.

This work, thus is essence of aforementioned researches in the field of mobile robotics. Due to need of modelling of vehicle, motion dynamics and controls are modelled for simulating our robot in MATLAB coupled with Simulink environment. As FURBOT is unique freight vehicle, modeling and simulating it is key for performance evaluation for vehicle and environmental safety.

2 Vehicle Dynamics

The current mathematical model of FURBOT is being developed for judging the performance of the vehicle before embedding any new hardware (sensors) or implementing any new automated technique which could result in an accident if not simulated prior to the experiments being conducted.

2.1 Constraints

The mathematical model is able to perform well on an empty straight road. Additionally, it is assumed that the vehicle does not have any inherent errors which can deviate it from the straight path e.g. tyre misalignment/balancing, lateral center of gravity shift etc. The road geometry consists of a straight road with null radius of curvature. The only varying feature of the road is the gradient of the road, which is extracted from Google Maps and simulated for traction power and braking control of the vehicle while driven on a hilly terrain.

2.2 Mathematical Modeling

An effort is made to make the mathematical model as precise as possible. The current coordinate system has its x-axis along the nose of the vehicle, y-axis towards left and z-axis is upwards [7]. It is considered that the vehicle is unable to roll, move along z-axis or pitch (unless due to road gradient change), making this mathematical model as a primarily three degree of freedom model. Both forces acting along x and y axis are modelled with moments acting along z-axis. However due to constraints, the motion of the vehicle is along a straight line.

Traction Force. Most of the definitions of forces acting along the body x-axis are taken from Ref. [8]. The forward force generated due to the torque acting on the driven wheels is given by Eq. 1

$$F_t = \frac{T_p \iota_g \iota_o \eta_t}{r_d} \tag{1}$$

Where F_t is the traction force and T_p is the torque output of the power plant and in our case is the output of the throttle controller. ι_g is the transmission gear ratio, ι_o is the final drive gear ratio, η_t is the final efficiency of the driveline from the wheels to the power plant and r_d is the radius of the wheels respectively.

Drag Force. The drag force calculation is straight forward and is estimated with the shape of FURBOT in mind. At present, drag coefficient (C_d) of 0.5 is selected which is in reference to usual drag coefficient of such shape vehicle. Equation 2 is the equation used for drag force calculation.

$$F_d = 0.5\rho V^2 A_f C_d \tag{2}$$

Where F_d is the drag force acting on the body, ρ is the air density, A_f is the vehicle frontal area and V is the total velocity of the vehicle.

Gradient Force. Although the gradient force calculation is comparatively simpler i.e. dependent only on weight of the vehicle M_v and gradient angle α (Eq. 3, [8]), but the amplitude of this force can be significantly higher compared to other resistive forces. Also since in our case, there is no traffic or hurdle, this is the only force which cause braking to come in action (in case of negative gradient).

$$F_g = M_v g sin\alpha \tag{3}$$

Rolling Resistance Force. Rolling resistance force is due to the friction between tyres of the vehicle and the surface of road. The two main components attributing to this force are the normal force acting on the vehicle tyres and rolling resistance coefficient. Calculation of normal force is pretty straight forward, however there are number of different ways to calculate rolling resistance coefficient specially with varying velocity of the vehicle. The main equation for rolling force is given by Eq. 4 given in Ref. [8].

$$F_r = (M_v g cos\alpha) f_r \tag{4}$$

Where f_r is the rolling force coefficient and remaining is the normal force acting on the vehicle. For calculation of f_r numerous techniques are found in literature, however for this work, calculation of rolling force coefficient is taken from the work of Brian [9] which is also an extension of his work in [10] and is given in Eq. 5.

$$f_r = C_{sr} + \{3.24 C_{dr} (\frac{V}{100})^{2.5}\} \tag{5}$$

Where C_{sr} and C_{dr} represents the static and dynamic components of rolling resistance coefficient. In [11], variation of both, C_{sr} and C_{dr} are plotted against tyre pressure and Brian [9] used these graphs to extract polynomial expressions for C_{sr} and C_{dr} which are given in Eq. 6 and 7. These equations are thus taken from the work of Brian [9] and their validity is discussed in his work.

$$(C_{sr})_{ref} = -0.0000001687 P_i^3 + 0.0000255349 P_i^2 - 0.0012944847 P_i + 0.0305104628 \quad (6)$$

$$(C_{dr})_{ref} = -0.0000002636 P_i^3 + 0.0000404822 P_i^2 - 0.0020812137 P_i + 0.0381150798 \quad (7)$$

Where P_i denotes the tyre pressure. The above equations are considered for calculating the rolling resistance force coefficient for this work.

Forces Along y-Axis and Moment Along z-Axis. The Newton-Euler equations of motion for forces along y-axis and moment along z-axis are used for calculating respective forces and moments and are given by Eq. 8 and 9 [12].

$$\dot{v_y} = \frac{1}{M_v v_x}(-a_1 C_{af} + a_2 C_{ar})r - \frac{1}{M_v v_x}(C_{af} + C_{ar})v_y + \frac{1}{M_v} C_{af}\delta - r v_x \quad (8)$$

$$\dot{r} = \frac{1}{I_z v_x}(-a_1^2 C_{af} - a_2^2 C_{ar})r - \frac{1}{I_z v_x}(a_1 C_{af} - a_2 C_{ar})v_y + \frac{1}{I_z} a_1 C_{af}\delta \quad (9)$$

These equations are expressed in the body coordinate frame for the planar rigid vehicle [7]. C_{af} and C_{ar} are the cornering stiffness of the front and rear wheels respectively. δ is the steering angle and a_1/a_2 are the distances of the rear/front wheels from the CG of the vehicle. Since our steering angle for this work is considered zero, the forces acting along y-axis and moments acting along z-axis yield very negligible values which are not enough for moving the vehicle considerably in y-axis.

Longitudinal Equation of Motion Along x-Axis. The longitudinal equation of motion is relatively simple as the problem is reduced to forces along x-axis and is given by Eq. 10.

$$\dot{v_x} = \frac{\sum F_x}{M_v} = \frac{F_t + F_d + F_g + F_r}{M_v} \quad (10)$$

2.3 Velocity Control

Two separate controllers are designed for motion control for the autonomous vehicle FURBOT. One for traction power control and one for braking power control. The reason for using such cascade controller is because our requirements for braking and acceleration are different. For traction power control, we require a smooth robust controller whereas for braking power controller, we require sharp responses addition to smooth behavior. Traction power control is a PD controller with an error amplification factor, whereas braking power control is a simple proportional error control with a self-defined operational dead-band of 1 km/h speed, thus it is only initiated if there is a difference of at least 1 km/h speed between reference and actual speed. Reason for not including integral component for the controller designs was because of the overshoot integral values were causing in actuation values. The designing criteria for both the controllers was to keep the velocity error < 2 km/h. Inputs of both controllers is the difference between reference and actual velocity in km/h. Details of these controllers are given in Table 1.

Table 1. Velocity controllers

Controller type	Controller values		
	Error amplification	Proportional gain	Derivative gain
Traction power control	500	20	1
Braking power control	1	80	0

A switch is placed between traction controller and braking controller. If the signal of braking control is greater than 0 i.e. it is active, the traction controller is automatically turned off. The controller design is kept simple in order to build the mathematical model.

3 Simulation Setup and Results

The whole mathematical model was built in MATLAB and SimuLink and simulated for obtaining outputs for validating the model under genuine conditions.

3.1 Simulation Setup

Genova, Italy has one of the most unique terrain topology. Not only does it have beaches and mountain ranges, it is also quite dense in population. Driving an autonomous vehicle on such a terrain for the first phase of testing requires extensive simulation testing for environment and vehicle safety. A potential route is selected in Genova for FURBOT for its performance evaluation. The route selected for the FURBOT mathematical model verification and for elevation data is taken from Google Maps [13] and is shown in Fig. 1

For results evaluation, only the elevation profile is extracted. The current scope of work is to evaluate FURBOT capability over hilly terrain thus the selected path served as an ideal candidate for performance evaluation. The elevation profile extracted from the Fig. 1 is plotted in Fig. 2.

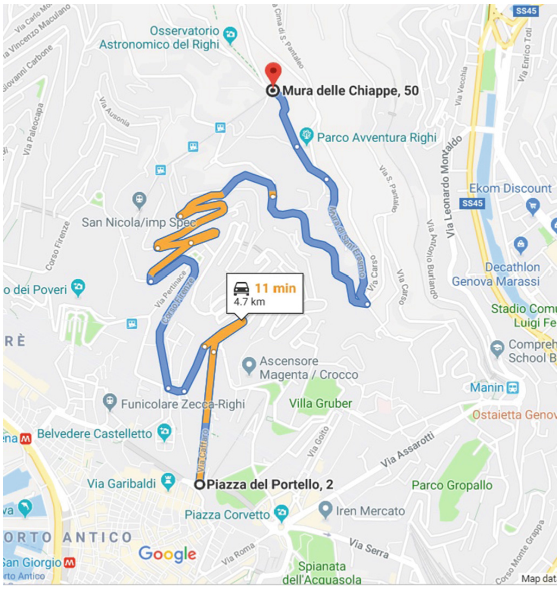


Fig. 1. Selected elevation route

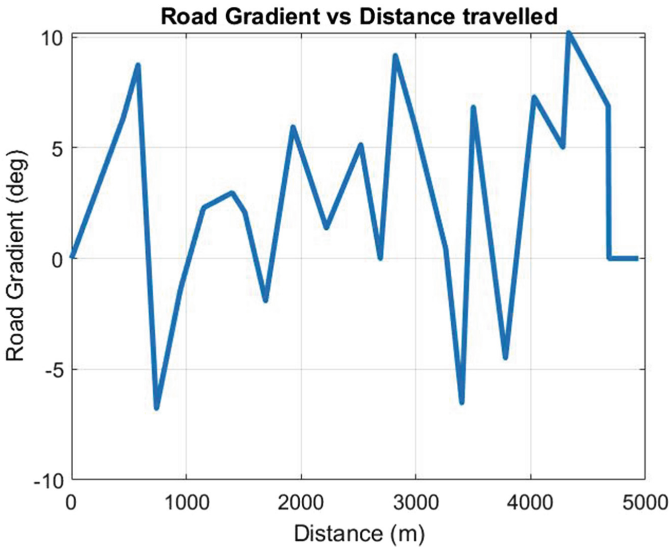


Fig. 2. Extracted elevation data

3.2 Results

In the current simulated scenario, the vehicle behaved comparatively well. The reference velocity was set to 40 km/h for the vehicle and velocity control contained the velocity error well within acceptable 1 km/h bound. Figure 3 shows the overall velocity of the vehicle and Fig. 4 shows the relative error in velocity in the whole simulation.

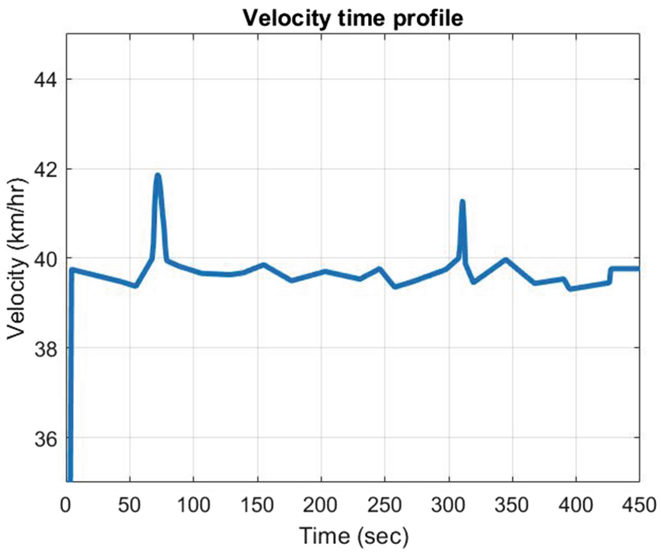


Fig. 3. Velocity profile over time

It is observed from the velocity profiles that it is easier for controlling the vehicle uphill compared to downhill as there are smooth transitions on positive gradient. The net effective traction force profile is given in Fig. 5. It is observed from the traction force profile that it follows the profile of the road gradient. Additionally, acceleration is zero in the negative gradient zone of the road which is also conceptually correct. Overall this shows satisfactory performance of the traction force controller.

The braking of the vehicle is triggered twice in the current scenario on the downhill journey of the vehicle. When compared with the gradient of the road, it shows coherence of braking with negative road gradient. A dead-band of 1 km/h is deliberately selected for avoiding any unnecessary use of braking. Figure 6 shows the comparison of braking force with the velocity error. This shows that braking is only triggered when higher than reference velocity is attained.

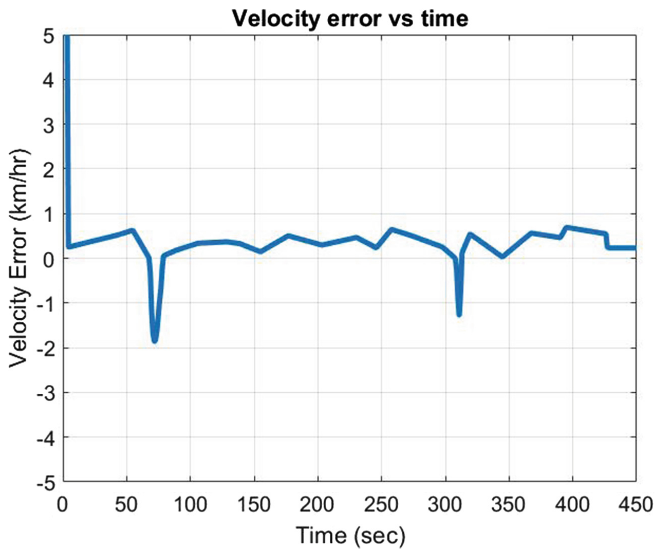


Fig. 4. Velocity error profile

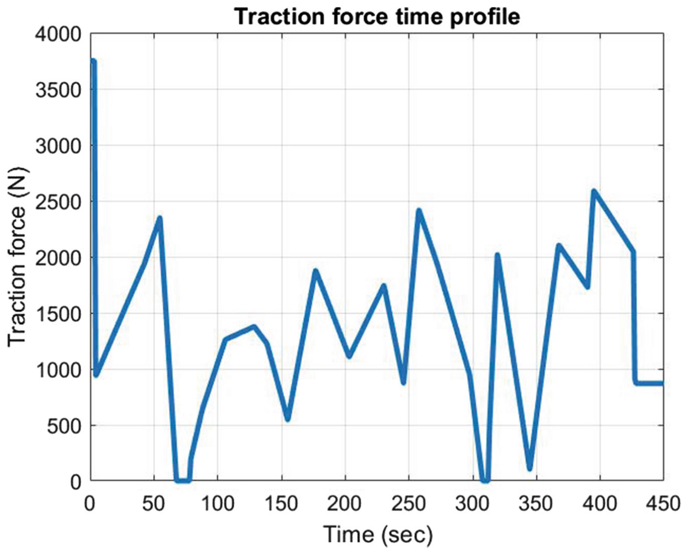
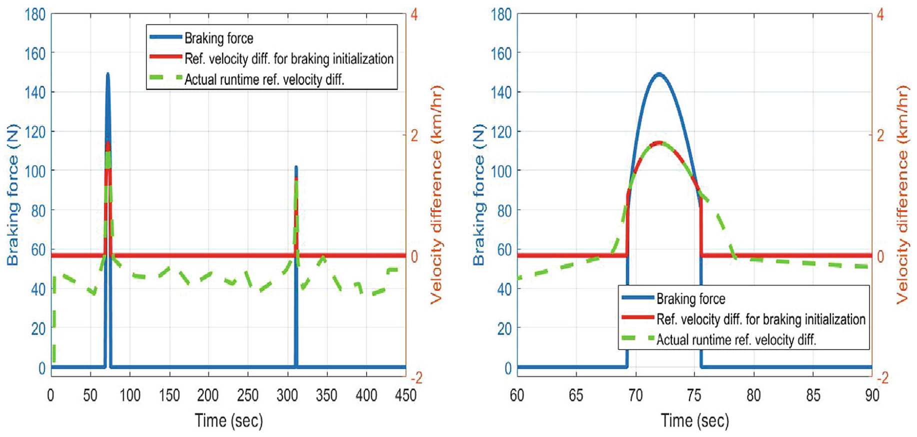


Fig. 5. Traction force profile

If we zoom-in Fig. 6 a) at one of the braking instances, as shown in Fig. 6 b), we can clearly observe the application of the velocity dead-band implemented for braking. Where green shows the velocity difference profile and red denotes the velocity difference profile input to the braking controller.



(a) Full profile (b) Zoomed profile

Fig. 6. Braking force and velocity error profile

4 Conclusion

The mathematical model for FURBOT worked as per requirements. The velocity control for the vehicle created nominal errors which were within tolerable range of 2 km/h. The switching between the cascade controller for velocity control also behaved as per need. The vehicle was able to maintain its velocity over the uneven hilly terrain which was the goal of the research. Furthermore, the whole mathematical model generated realistic results.

After modeling complete road topology and embedding traffic data into the simulation, steering control can be incorporated. This can make the vehicle's mathematical model complete and autonomous which will be of critical importance for selecting and testing new sensors for the vehicle. This model will additionally serve as a platform for the future work on this autonomous vehicle. Number of safety enhancements can be incorporated in the vehicle after simulating its behavior. Some future outputs of this system include path planning, safe parking, cargo alignment and enhancing the safety of the vehicle and the environment which includes bounds on top speed, radial velocity, minimum safe distance and operational battery life before recharge is required.

Acknowledgment. This paper is supported by European Union's Horizon 2020 research and innovation programme under grant agreement No. 875530, project SHOW (SHared automation Operating models for Worldwide adoption).

References

1. Westervelt, E.R., Grizzle, J.W., Chevallereau, C., Choi, J.H., Morris, B.: Feedback control of dynamic bipedal robot locomotion. CRC Press, Boca Raton (2018)

2. Fiore, E., Giberti, H., Ferrari, D.: Dynamics modeling and accuracy evaluation of a 6-DoF Hexaslide robot. In: Nonlinear Dynamics. Conference Proceedings of the Society for Experimental Mechanics Series, vol. 1, pp. 473–479 (2016). https://doi.org/10.1007/978-3-319-15221-9_41
3. Pedrammehr, S., Qazani, M.R.C., Abdi, H., Nahavandi, S.: Mathematical modelling of linear motion error for Hexarot parallel manipulators. Appl. Math. Model. **40**(2), 942–954 (2016). https://doi.org/10.1016/j.apm.2015.07.004
4. Esmaeili, N., Alfi, A., Khosravi, H.: Balancing and trajectory tracking of two-wheeled mobile robot using backstepping sliding mode control: design and experiments. J. Intell. Robot. Syst. **87**(3–4), 601–613 (2017). https://doi.org/10.1007/s10846-017-0486-9
5. Asano, F., Seino, T., Tokuda, I., Harata, Y.: A novel locomotion robot that slides and rotates on slippery downhill. In: 2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM) (2016). https://doi.org/10.1109/aim.2016.7576804
6. Rodriguez, R., Ardila, D.L., Cardozo, T., Perdomo, C.A.C.: A consistent methodology for the development of inverse and direct kinematics of robust industrial robots. J. Eng. Appl. Sci. **13**(1), 293–301 (2018)
7. Marzbani, H., Khayyam, H., To, C.N., Quoc, D.V., Jazar, R.N.: Autonomous vehicles: autodriver algorithm and vehicle dynamics. IEEE Trans. Veh. Technol. **68**(4), 3201–3211 (2019). https://doi.org/10.1109/tvt.2019.2895297
8. Ehsani, M., Gao, Y., Longo, S., Ebrahimi, K.: Modern Electric, Hybrid Electric, and Fuel Cell Vehicles. CRC Press, Taylor & Francis Group, Boca Raton (2019)
9. Wiegand, B.P.: Estimation of the Rolling Resistance of Tires. SAE Technical Paper Series (2016). https://doi.org/10.4271/2016-01-0445
10. Wiegand, B.P.: Mass Properties and Advanced Automotive Design. SAWE Technical Paper 3602, 74th SAWE International Conference on Mass Properties Engineering; Alexandria, VA (2015)
11. Dixon, J.C.: Suspension Geometry and Computation. John Wiley & Sons Ltd., Chichester, UK (2009). ISBN 978-0-470-51021-6
12. Fu, C., Hoseinnezhad, R., Bab-Hadiashar, A., Jazar, R.N.: Electric vehicle side-slip control via electronic differential. Int. J. Veh. Auton. Syst. **6**, 1–26 (2014)
13. Google (n.d.): Google Maps directions for driving from Piazza del Portello, Genova to Righi, Genova. https://www.google.com/maps/dir/44.4114759,8.9345774/44.4241951,8.9379112/@44.4185213,8.9331592,15z/data=!4m2!4m1!3e0?hl=en. Accessed 12 Sept 2019