



Kinematic Modeling for Biped Robot Gait Trajectory Using Machine Learning Techniques

Bharat Singh¹ · Ankit Vijayvargiya¹ · Rajesh Kumar¹

Received: 26 July 2021 / Revised: 7 December 2021 / Accepted: 14 December 2021 / Published online: 19 January 2022
© Jilin University 2022

Abstract

This paper presents the predictive models for biped robot trajectory generation. Predictive models are parametrizing as a continuous function of joint angle trajectories. In a previous work, the authors had collected the human locomotion dataset at RAMAN Lab, MNIT, Jaipur, India. The MNIT gait dataset consists of walking data on a plane surface of 120 human subjects from different age groups and genders. Thirty-two machine learning models (linear, support vector, k-nearest neighbor, ensemble, probabilistic, and deep learning) trained using the collected dataset. In addition, two types of mapping, (a) one-to-one and (b) many-to-one, are presented for each model. These mapping models act as a reference policy for the control of joints and prediction of state for the next time instant in advance if the onboard sensor fails. Results show that the deep learning and probabilistic learning models perform better for both types of mappings. Also, the probabilistic model outperforms the deep learning-based models in terms of maximum error, because the probabilistic model effectively deals with the prediction uncertainty. In addition, many-to-one outperforms the one-to-one mapping because it captures the correlation between knee, hip, and ankle trajectories. Therefore, this study suggests a many-to-one mapping using the probabilistic model for biped robot trajectory generation.

Keywords Kinematic · Locomotion · Machine learning · Prediction · Regression

1 Introduction

In recent times, robots have found numerous applications in real time such as picking objects, walking, path tracking, disaster management, health care, automation industries, house works, space, and entertainment [1]. Numerous types of robots based on air, water, and land-based application are found in the literature [2–5]. In this paper, the authors have mainly focused on land-based legged robots. The legged robot can be a potential candidate to use in an unstructured real-world environment. The legged robot-like biped robot can mimic human locomotion and outperform the other robots in the modern infrastructure that means for humans

[6, 7]. However, current research in biped robots is far from a real-time implementation, because of the requirement of firm control to remain stable while navigating through the stochastic environment [8]. Since biped robots have numerous degrees of freedom, it enables them to perform various tasks. However, even for fundamental tasks like walking, there is the necessity for planning and continuous trajectory generation of the complete body which is very difficult in general. To realize such a complex movement, a hierarchical structure of a control framework is required. The higher-level part provides the reference trajectory, whereas the low-level part generates the decision to track this reference trajectory under disturbances robustly, and precision control is required to achieve fast motion.

In literature [9–16], different approaches have been developed and successfully applied for the versatile locomotion of biped robots. The inverted pendulum is the most predominant technique used to generate the reference trajectory [11, 12]. This topology provides the center of mass (CoM) and foot trajectories, which can be transferred to the whole robot using inverse kinematics. However, obtaining a detailed description of any physical system is a very

✉ Bharat Singh
bharatsingh1993@yahoo.com
Ankit Vijayvargiya
ankitvijayvargiya29@gmail.com
Rajesh Kumar
rkumar.ee@mmit.ac.in

¹ Department of Electrical Engineering, Malaviya National Institute of Technology, Jaipur 302017, India

cumbersome task. Therefore, the models derived from this method can introduce bias because of parameter variation and unmodeled non-linear terms. Likewise, the multi-objective evolution of trajectory synthesis based on ant colony-optimized recurrent neural networks has also been developed [17]. However, the trajectory is not matched with humans, because humans have locomotion capability naturally. Therefore, humans can perform a lot of diverse and complex tasks efficiently, still unmatched by current state-of-art biped robots. As per literature [18, 19], researchers have developed various methodologies for transferring the human motions to the biped robots and tested them in a constrained environment. An inverse optimal control methodology employs finding the optimal criteria from human locomotion [19]. The authors develop a 3D template model based on the optimal criteria. It describes the motion based on CoM trajectory, foot trajectory, upper-body orientation, and time duration, which is transferred to the full body using inverse kinematics. In reference [18], the authors have introduced a reduced-order data-driven model based on human locomotion data. The joint kinematics is parameterizing as a function of joint angles. The goal is to predict the kinematic gait pattern with fewer data. However, it involves complex mathematics to be solved, which is very cumbersome. Likewise, the authors have directly transferred the joint angle trajectories to robots [20]. However, there is a difference in physical constraints between humans and robots. So, it may not be directly feasible to apply the joint trajectories on real robots. Mostly, these model-based methods are promising topologies to extract valuable information from the available data. However, these model-based methods require more accurate knowledge of robot dynamics [21].

Since several researchers have focused on the regression model because of the availability of data, it minimizes the requirement of the exact mathematical expression for robot dynamics. As per literature, the regression models are classified into deterministic, uncertainty, and learning category. Figure 1 illustrates the classification of the regression models. During the last decade, there has been an enormous increment in machine learning applications, since the machine learning algorithms have become accurate and their capability has significantly improved. This improvement is associated with a resurgence of computing power. It allows in using various machine learning algorithms, neural networks, and deep learning, and the requirement of dynamic knowledge of robots has been significantly reduced.

In this paper, the authors present 32 machine learning models for the mapping of joint kinematics. These models serve two purposes: (a) reference trajectory and (b) prediction for the next time step task. Case I: reference trajectory for the policy is generated by the one-to-one

mapping (knee \rightarrow knee, hip \rightarrow hip and, ankle \rightarrow ankle) and many-to-one mapping (knee + hip + ankle \rightarrow knee, knee + hip + ankle \rightarrow hip and, knee + hip + ankle \rightarrow ankle). Case II: prediction for the next time step task is explained as follows. Assume a situation when the onboard sensor fails to provide the feedback signal to the policy as shown in phase I of the biped learning framework flowchart (Fig. 2) or a reduction in the number of sensors for cost saving. Then, the policy cannot get the feedback signal (i.e., the joint angle at that instant). In this situation, the robot can go toward instability and subsequently lead to permanent damage of hardware. Therefore, the trajectory generation model can be used to provide feedback to a policy shown in phase II of the biped learning framework flowchart (Fig. 2). This trajectory generation model has a mapping of many-to-one. The important note here is that the next state is estimated based on the current state in advance. So, the policy already has the required missing state for the next time instant.

As per the author's knowledge, there is not much literature/study found on the kinematic modeling of locomotion data using machine learning techniques. Therefore, in this paper the authors have analyzed the 32 learning techniques which help in finding a suitable approach for kinematic modeling. In addition to those, two types of mapping are also developed. The major contributions of the presented work are:

- Machine learning techniques are applied to develop the kinematic model using human locomotion data.
- Identification of suitable learning technique for kinematic modeling based on performance indices, i.e., average error, maximum error, and root mean square.
- Two types of mapping, (a) one-to-one and (b) many-to-one, are developed for reference trajectory generation and prediction for the next time step task in advance.

The taxonomy of the paper is as follows: Sect. 2 describes the data collection process; various machine learning techniques such as linear models, support vector machines, nearest neighbor, ensemble learning, probabilistic, and deep learning approaches; and various performance indices. Section 3 describes the results of various developed models for one-to-one and many-to-one mapping. Finally, Sect. 4 presents the conclusion and future research direction.



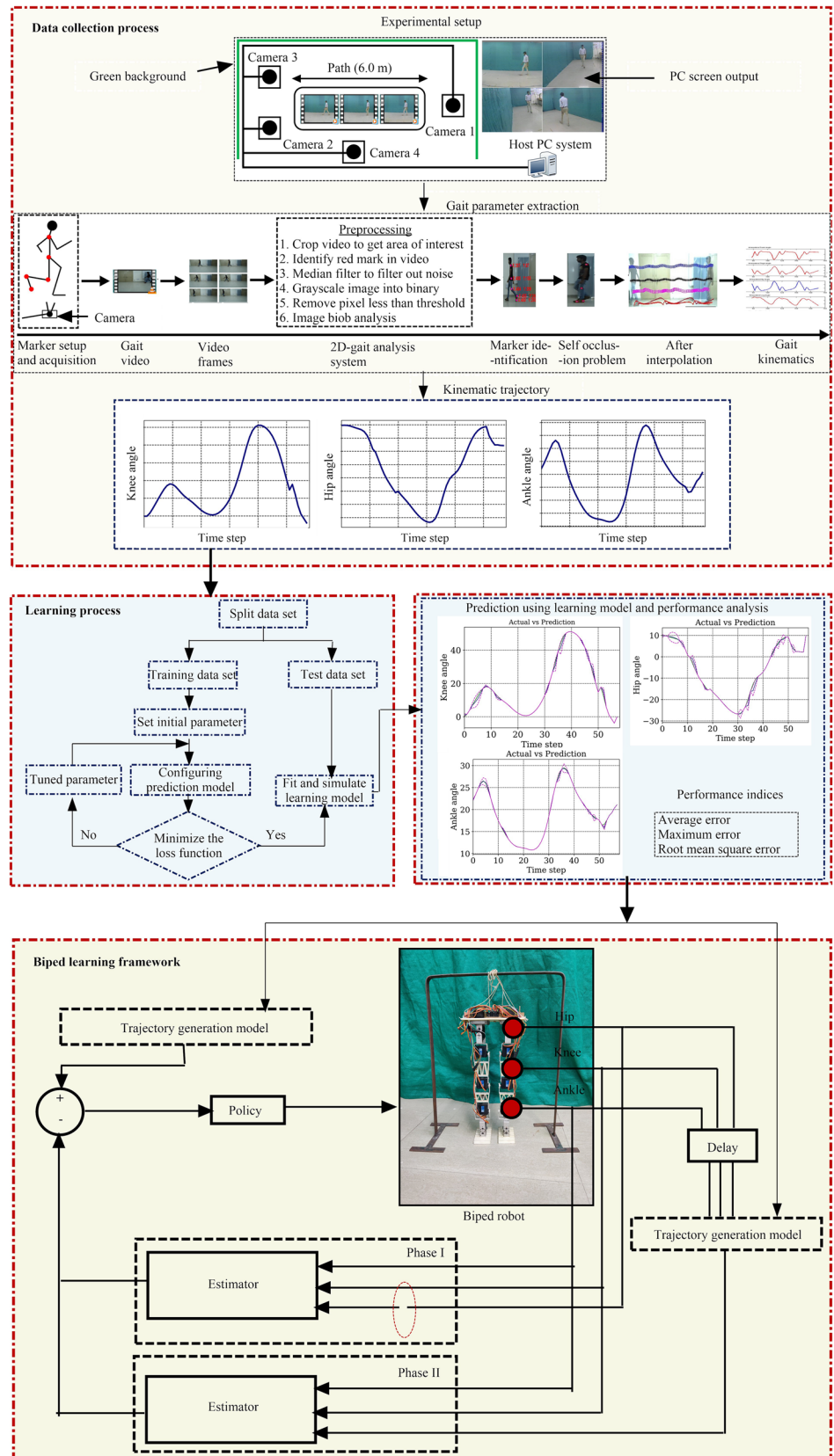
Fig. 1 Classification of regression models

2 Proposed Methodology

This section discusses the data collection process, different machine learning techniques, and performance evaluation indices. Figure 2 illustrates the general flowchart of methodology. The basic step of methodology is given as follows:

- Collect the human locomotion data.
- Perform data preprocessing and standardization of the collected data.
- After the above step, split the data into two parts: (a) training dataset and (b) testing dataset.
- Fit the learning models for the training dataset and examine the loss value.
- Test the accuracy of the learned model using the testing dataset.
- Calculate the performance indices of the learned models.

Fig. 2 Proposed flowchart



- Implement the learned models on the biped robot for trajectory generation.

2.1 Data Collection

In previous works [22–29], the authors carried out a gait analysis and collected the human locomotion cycle data using vision-based techniques at Robotics and Machine Analytics Laboratory (RAMAN Lab), MNIT Jaipur, India. Data consist of the kinematic trajectory of the hip, knee, and ankle angle trajectory. The experimental setup has a digital camera, which provides the recording of locomotion video for data acquisition to the personal computer (PC) and processes it. Digital video recorder, Nikon D5200 DSLR, and BIOPAC® 4 cameras are used to capture the motion in the sagittal plane at 29, 50, and 30 fps, respectively. Setup for gait analysis is presented in the data collection process of Fig. 2. The green color background is used for data collection. The walking path for every subject is chosen equal to 6 m which provides enough space to take about four steps. Subjects walk at a normal pace from right to left on walking and then left to right. The MNIT gait dataset is developed from 120 subjects from different age groups and genders. The age of the subjects is in range of 5–60 years. All the subjects had no past injury or pain during data collection. Here, the walking platform is plane (0-degree inclination). Data preprocessing is done using the gait analysis model that is developed in simulation [20, 21]. MATLAB and python programming languages are used to implement and simulate the gait analysis process. Most gait capture system uses a direct measurement technique to capture the required features; however, the cost for hardware and natural motion of human is hindered. Thus, one of the main challenges is to reduce the number of sensors required on the subject body. Therefore, two methods have been proposed for gait capture to overcome the major challenges: (A) passive marker-based gait parameter extraction approach (PM-GPEA)—Model 1 [24] and (B) markerless/holistic-based gait parameter extraction approach (MI-GPEA)—Model 2 [29].

2.2 Machine Learning Techniques

In this section, the authors have explored the different machine learning models for the development of mapping models. The state of assumed prediction model at time instant t is given by Eq. (1),

$$\delta y_t = f(y_t). \quad (1)$$

Therefore, prediction for time instant $t+1$ is given by Eq. (2),

$$y_{t+1} = y_t + \delta y_t. \quad (2)$$

2.2.1 Linear Models

Linear models are based on the statistical approach for regression analysis. Two types of variables are defined, namely (a) the dependent variable is influenced by the target variable, and (b) the independent variable not influenced by the input variable. In simple linear regression, dependent variables have only one variable which can be used in regression analysis, whereas the independent variables always have more than one. Thus, the target value is composed of a linear combination of different features (in our case, it is a combination of the knee, ankle, and hip angle).

2.2.1.1 Linear Regressor It minimizes the sum of squares between the observed targets in the dataset and the output predicted by the model [30]. However, in most cases, the nature of the predicted target is not linear. Hence, in this paper, the authors have used polynomial fitting.

2.2.1.2 Ridge Regressor Linear regression is prone to overfitting due to its structure. Therefore, it is necessary to add the additional term to the minimization function known as regularization [31].

2.2.1.3 Lasso Regressor The solution obtained from Support Vector Regression (SVR) is not sparse in nature i.e., the percentage of support vectors is high and they are highly correlated to each other. Therefore, an alternative approach, namely the least absolute shrinkage and selection operator (Lasso) regression is developed [32]. It provides sparse solutions because of using the L_1 norm in minimization of error. Basically, it aims to identify the parameters that can minimize the prediction error. This is obtained by imposing the constraint on a model by applying the shrinkage on regression parameters toward zero.

2.2.1.4 Lars Regressor It is a model selection method like stepwise regression [33]. In Lars method, all the possible Lasso estimates are evaluated for a given objective function. At every time step, it finds the most correlated input features with the target.

2.2.1.5 Hybridization of Lasso and Lars Regressor It includes the advantages of both Lasso and Lars regression, i.e., a sparse solution with L_1 norm and model selection [34]. Lasso model integrated with Lars using BIC (Bayes information criterion) and AIC (Akaike information cri-

terion). This criterion helps in selecting the value of the regularization parameter based on the trade-off between the fitness and complexity of the model.

2.2.1.6 Huber Regressor Huber regressor is developed to deal with outliers by introducing the least square penalty for small residues. The idea is to use the different loss functions rather than the ordinary least squares [35]. This function behaves like a least squares penalty for small residues. However, the impact is low for the large residues. Lead to increment linearly rather than quadratic.

2.2.1.7 RANSAC Regressor It is a non-deterministic algorithm that produces a reasonable solution with a certain probability. It depends on several iterations to produce a robust solution. It divides the data sets into the set of inliers and outliers. Here, inliers are used in iterations for the estimation of the solution [36].

2.2.1.8 TheilSen Regressor The problem associated with the Ordinary Least Square (OLS) method is that the efficacy reduces when an error is normal and heteroscedastic. Thus, the slope of linear regression remains in an unsatisfactory probability range and the problem becomes significant if the error is non-normal. Theil and Sen developed a new method to deal with these problems in 1950 and 1968, respectively. It is a robust technique that fits the model to sample points in the space of a plane, and chooses the median of slopes in-plane (various lines) through the selected pair of points. The slope is known as Sen's slope. This makes the estimator robust toward the outliers. It also makes the model more efficient [37].

2.2.1.9 ElasticNet Regressor Imposition of L_1 norm in error minimization provides a sparse solution. Thus, it also provides both shrinkage and automatic variable selection simultaneously. However, due to its nature of convex optimization problem, Lasso selects at most n variables before it saturates if several predictors (p) > the number of observations (n). Also, it performs well only if the bound on the parameter for the L_1 norm is smaller than a certain threshold value. Therefore, a new regularization technique is developed [38], which has similar properties to Lasso. It has additional advantages like the grouping effect where strongly correlated predictors are selected in groups. It is like a stretchable fishing net that consists of all big fishes.

2.2.1.10 Orthogonal Matching Pursuit Regressor This is a recursive method that evaluates the functions with respect to the non-orthogonal direction. The matching pursuit

algorithm is updated such that it brings the full backward orthogonality of the residual convergence [39]. Computation is performed recursively.

2.2.2 Support Vector Machines

2.2.2.1 Kernel Ridge Regressor Kernel ridge regressor is a simplified case of SVR coined by Cristianini and Shawe-Taylor in 2000 [40]. It is a non-parametric form of ridge regression. Here, all data points are transformed with their feature vector as $x_i \rightarrow k_i = K(x_i)$. Thus, it learns a function in space of kernel by minimization of squared error with a squared norm regularization term.

2.2.2.2 Support Vector Regressor The idea of Support Vector Machine (SVM) is derived from the work of Vapnik and Learner (1963) and Vapnik and Chervonenkis (1964), which is a non-linear generalization of generalized portrait algorithm. Basically, Support Vector Regression (SVR) is the generalization of SVM which is accomplished by integrating the ϵ -insensitive region around the function called ϵ -tube [41, 42].

2.2.3 K-Nearest Neighbors

It is a non-parametric machine learning algorithm [43]. The principle of KNN is that similar samples have the highest probability for the same class. Basically, it collects all the samples, and based on the similarity index predicts the target value.

2.2.4 Ensemble Learning Models

2.2.4.1 Decision Tree Regressor In designing Decision Tree Regressor (DTR), one feature is selected at each node and partitions made in the range of that feature to produce a smaller number of bins, where each bin is termed as a child of the node. When the features' range is discrete with N unordered outcomes, the partitions are done by minimizing an expected loss function that is usually found by exhaustive search through all partitions [44]. The advantage of using the DTR is that it requires less effort for data preparation during the preprocessing, no requirement for the normalization of data or the scaling of data, and missing values in data do not affect the building process of the tree. However, it requires a longer time for the training of the model.

2.2.4.2 Random Forest Regressor The idea of random forests was floated by Leo breiman in 2000s. Here, the predictor ensemble is built with a different set of decision trees

that grows from a random subset of data. Generalized error converges to a limit as the size of the tree becomes large. However, the error of every tree in the random tree depends on the strength of the individual tree and the correlation present between them [45]. However, it can be biased while dealing with categorical variables, requires more training and is not suitable for linear methods with a lot of sparse features.

2.2.4.3 Adaptive Boosting Regressor Boosting is a methodology that combines various weak and high error learning techniques. It can provide strong and low error techniques. Thus, it can be used to reduce the error of less accurate learning techniques, Basically by running repeatedly the learning approach on a different distribution of training data and outputs. In each loop of training, the distribution of training data depends on the previous loop performance of the learning technique. One important thing is that for various boosting approaches, the method that evaluates the different distribution of training data and combination of predictions is different. The generalization of Adaboost is given by schapire, which provides an interpretation of boosting as a gradient-descent method [46]. The algorithm uses a potential function that associates a cost. Operation of Adaboost can be described as coordinate-wise gradient descent in the space of weak learning topology.

2.2.4.4 Gradient Boosting Regressor In Adaboost, various weak learning algorithms are integrated together to boost the overall performance of an algorithm. However, it does not easily generalize to regression problems. It is because regression problems have faced difficulty due to hypotheses that may not be just wrong or right. However, it can be less right or wrong. In this scenario, the training error is more. Thus, a new algorithm is developed, namely gradient boosting regressor, where a new hypothesis is formed simply to modify the distribution of training data [47]. It aims to fit a

new predictor in the residual errors committed by the preceding predictor. The overall arrangement makes this algorithm less prone to overfitting.

2.2.5 Probabilistic Models

2.2.5.1 Bayesian Linear Regressor Bayesian linear regression uses the posterior probability distribution over the parameters instead of point estimation. It helps in dealing with the model bias. It signifies that the output response is not a single point estimate, and a response is drawn from the probability distribution. The goal is to find the posterior distribution over the parameters rather than the single best value [48, 49]. Likewise, Automatic Relevance Determination (ARD) is used to find the significance of elements in input to determine the response by assigning the shrinkage prior to every parameter.

2.2.5.2 Gaussian Process Regressor It puts a probability distribution over the latent functions. It determines the shape of latent function i.e., from the available data. It has major advantages like it directly captures the uncertainty present in the model and any prior information can be added in form of a prior distribution [50, 51]. However, it is so powerful that it can even fit the noise.

2.2.6 Neural Network Models

2.2.6.1 Multilayer Perceptron Regressor It provides the generalization in addressing the problems with non-linear shapes and has the capacity to approximate unknown functions. Generalization of perceptron using the multiple layers and connecting all neurons in one layer to all neurons of next layers gives rise to feedforward multilayer perceptron [52, 53]. Basically, input features are mapped to output using the network. It also allows the non-linear regression.

2.2.6.2 Deep Neural Network Regressor Since 2006 when Hinton proposed structured learning known as deep belief networks (DBN), various architectures are developed [54, 55]. DBNs are integrated with many stochastic and latent parameter variables. In the first instance, Hinton's group participated in the competition and shows that architecture results in 10% better results than the second participator model. Generally, the architecture of a deep neural network is hierarchical with many layers and each layer consists of a non-linear activation function.

Table 1 Commonly used performance indices for regression problems

S. no	Index	Mathematical expression
(a)	Average error (AE)	$\frac{\sum_{i=1}^N P_{r,i} - P_{p,i}}{N}$
(b)	Maximum error (ME)	$\max P_{r,i} - P_{p,i} $
(c)	Root mean square error (RMSE)	$\sqrt{\frac{1}{N} \sum_{i=1}^N P_{r,i} - P_{p,i} ^2}$

Table 2 Summary statistics of linear models

Model	P.I	One-to-one			Many-to-one		
		A	B	C	D	E	F
Ridge	AE	3.0485	1.5378	1.0447	0.6075	1.2033	0.8307
	ME	6.7596	3.4677	2.5741	1.8348	2.1497	1.9694
	RMSE	3.5564	1.9091	1.2434	0.7936	1.3921	0.9694
ElasticNet	AE	3.0444	1.5587	0.9732	0.8127	1.1652	0.7934
	ME	6.5743	3.4711	2.7253	2.1296	2.1378	1.6635
	RMSE	3.5457	1.9144	1.2062	1.0166	1.3418	0.9078
Huber	AE	3.0555	1.5372	1.0807	0.4831	0.9979	0.8558
	ME	7.1643	3.6546	2.7311	2.1410	2.0673	2.2099
	RMSE	3.5996	1.9352	1.3044	0.7368	1.1462	1.0339
RANSAC	AE	3.7871	1.5749	1.1667	0.5788	1.1819	1.3713
	ME	11.415	4.8798	3.7503	2.5085	3.6238	3.4566
	RMSE	5.0174	2.2163	1.5832	0.8837	3.4566	1.8618
TheilSen	AE	3.1381	1.6244	1.0565	0.9023	1.0089	0.7955
	ME	6.4188	3.7201	2.8397	2.2232	2.2922	1.6701
	RMSE	3.6429	1.9587	1.2957	1.1578	1.2191	0.9689
OMP	AE	3.0343	1.5773	0.9743	3.0421	1.3451	0.8735
	ME	6.4309	3.4799	2.7251	5.1502	2.3374	2.6142
	RMSE	3.5371	1.9242	1.2071	3.3957	1.5523	1.1531
Lasso	AE	3.0439	1.5761	0.9721	0.9301	1.1544	0.8101
	ME	6.4012	3.4735	2.7256	2.2717	2.2322	1.8908
	RMSE	3.5387	1.9229	1.2054	1.1423	1.3431	0.9337
Lars	AE	3.0485	1.5378	1.0481	0.6061	1.2042	0.8311
	ME	6.7601	3.4677	2.5672	1.8303	2.1515	1.9715
	RMSE	3.5565	1.9091	2.5672	0.7911	1.3932	0.9701
LassoLars	AE	3.0917	1.5465	0.9469	0.6062	1.2078	0.8310
	ME	6.2647	3.3473	2.7469	1.8303	2.1748	1.9715
	RMSE	3.5507	1.8967	1.1603	0.7910	1.3932	0.9700
LassoLarsIC	AE	3.0917	1.5465	0.9469	0.6061	1.3736	0.8311
	ME	6.2647	3.3473	2.7469	1.8303	2.4397	1.9715
	RMSE	3.5507	1.8967	1.1603	0.7911	1.5471	0.9700
Linear	AE	3.0485	1.5378	1.0481	0.6062	1.2042	0.8311
	ME	6.7601	3.4677	2.5672	1.8304	2.1515	1.9715
	RMSE	3.5565	1.9091	1.2455	0.7910	1.3932	0.9701

Table 3 Summary statistics of support vector machine

Model	P.I	One-to-one			Many-to-one		
		A	B	C	D	E	F
SVR	AE	3.1441	1.5948	1.0841	2.2599	0.6529	0.5251
	ME	7.1569	4.3694	3.0394	3.9288	2.0625	1.7701
	RMSE	3.7084	2.0977	1.4135	2.6294	0.9141	0.7416
LinearSVR	AE	3.8353	6.2521	1.8168	1.4806	0.9379	0.8519
	ME	11.92875	24.3706	4.4386	2.9191	2.5444	2.1989
	RMSE	5.0947	9.2241	2.3498	1.7311	1.2124	1.0821
NuSVR	AE	2.9994	1.5197	0.9891	2.3688	0.7078	0.6027
	ME	6.2751	3.6752	2.6223	4.3166	1.9324	1.3671
	RMSE	3.5031	1.9194	1.2137	2.6809	0.9614	0.6966
Kernel ridge	AE	3.0495	1.5371	1.0089	2.9039	1.2631	0.8481
	ME	6.7521	3.4671	2.6409	4.7172	2.9951	1.7721
	RMSE	3.5572	1.9087	1.2246	3.1768	1.6502	0.9521

Table 4 Summary statistics of K-nearest neighbor

Model	P.I	One-to-one			Many-to-one		
		A	B	C	D	E	F
Weight distance	AE	3.8592	1.7357	1.3342	1.3746	0.5518	0.3291
	ME	9.5046	4.5501	2.9735	4.3981	1.3736	1.2799
	RMSE	4.7576	2.2861	1.5971	1.7829	0.7121	0.4812
Weight uniform	AE	3.1085	1.6648	1.1337	1.6759	0.7213	0.4131
	ME	6.4051	4.2995	2.8593	4.4378	2.0459	1.2511
	RMSE	3.7021	2.1916	1.4132	2.1537	0.9371	0.5588

Table 5 Summary statistics of ensemble models

Model	P.I	One-to-one			Many-to-one		
		A	B	C	D	E	F
DTR	AE	5.0642	1.8739	1.7563	0.9634	0.7967	0.5576
	ME	12.0656	5.5479	3.5321	4.0845	3.1819	2.5351
	RMSE	6.4416	2.5975	2.0755	1.4061	1.6072	0.8494
AdaBoost	AE	4.3505	1.7429	1.4186	1.2237	0.6253	0.5945
	ME	11.7267	4.2817	3.2271	2.6019	1.2625	2.4662
	RMSE	5.6607	2.1297	1.7044	1.4221	0.7359	0.8739
Gradient boost	AE	4.8565	1.8314	1.6601	0.8646	0.5909	0.4582
	ME	11.5107	5.5855	3.2411	2.9906	1.3409	1.5291
	RMSE	6.0941	2.4899	1.9531	1.1442	0.7669	0.6291
Random forest	AE	3.8664	1.7988	1.4953	1.2836	0.5096	0.4513
	ME	6.4051	8.5176	4.8661	3.0942	1.2597	1.1365
	RMSE	4.6923	2.3886	1.8081	1.5391	0.6521	0.5759

Table 6 Summary statistics of probabilistic models

Model	P.I	One-to-one			Many-to-one		
		A	B	C	D	E	F
Bayesian ridge	AE	3.0552	1.5663	0.9578	0.6674	1.0934	0.7973
	ME	6.2968	3.4257	2.7292	1.9461	2.2268	1.7871
	RMSE	3.5373	1.9139	1.1929	0.8722	1.2656	0.9266
ARD	AE	3.0917	1.5465	0.9471	0.6541	1.2235	0.8061
	ME	6.2647	3.3473	2.7501	1.9488	2.1897	1.8685
	RMSE	3.5507	1.8967	1.1647	0.8555	1.4199	0.9399
GPR	AE	3.0438	1.5494	0.9449	1.1268	0.8751	0.1685
	ME	6.0495	3.3209	2.3646	3.3144	1.1231	0.8273
	RMSE	3.5151	1.8957	1.1596	1.5109	1.3281	0.2549

2.2.6.3 Deep RNN Regressor In a deep Recurrent Neural Network (RNN), the recurrent structure is obtained by the integration of multiple layers of non-linear hidden layers [56]. It allows the temporal dynamic behavior. At every time step t , the input of the first recurrent layer is external, while the subsequent layers in network is fed from the activation function of the previous one. It can process inputs of any length and the model size does not increase as well. It uses the internal memory for processing the arbitrary series of

inputs which is not the case with feedforward neural networks.

2.2.6.4 Deep LSTM Regressor The simple recurrent network did not scale to long-term temporal dependencies. Thus, RNNs with Long Short-Term Memory (LSTM) was developed by Graves and Schmidhuber [57]. It deals with problems related to sequential data. Hence, these networks are scalable to long-term dependences. The central theme of

Table 7 Summary statistics of deep learning models

Model	P.I	One- to-one			Many-to-one		
		A	B	C	D	E	F
MLP	AE	6.2908	1.9121	5.5197	0.7508	0.2908	0.4255
	ME	23.7771	4.7296	11.2024	2.1772	1.0202	0.9461
	RMSE	9.4732	2.3151	6.36319	1.0395	0.4409	0.4999
DeepNN	AE	2.8857	1.4954	0.7807	1.2021	0.4227	0.4001
	ME	6.5401	3.309	2.1124	6.7663	1.4643	2.4318
	RMSE	1.6987	1.2228	0.8835	1.0963	0.6501	0.6325
RNN	AE	3.0781	1.7569	1.1838	0.9465	0.4091	0.5324
	ME	6.0731	3.6246	1.3255	5.7527	0.8457	3.7236
	RMSE	1.7544	2.8473	1.0881	0.9272	0.63962	0.7296
LSTM	AE	3.1862	1.6079	0.91553	0.9717	0.6979	0.2851
	ME	6.2777	3.4711	2.4823	7.4635	2.4919	1.6657
	RMSE	1.7851	1.2681	0.9568	0.9857	0.8354	0.5339
GRU	AE	2.6207	1.5868	1.4354	0.5615	0.4323	0.1594
	ME	7.7649	5.5331	3.0667	5.9958	1.3107	0.7497
	RMSE	1.6188	1.2596	1.1981	0.7493	0.6575	0.3993
BiRNN	AE	2.6454	1.4872	1.0455	1.5585	0.4636	0.7678
	ME	6.3375	4.4308	2.8836	4.6074	1.4691	3.3267
	RMSE	1.6264	1.2195	1.0224	1.2484	0.6808	0.8762
BiLSTM	AE	2.6951	1.7883	1.117	0.3751	0.5196	0.3314
	ME	7.3271	4.3861	2.8323	2.4749	2.8441	1.9941
	RMSE	1.6416	1.3372	1.0543	0.6124	0.7208	0.5757
BiGRU	AE	2.5379	1.3471	1.2121	0.4399	0.4585	0.3701
	ME	6.3704	3.93727	2.2754	5.1721	2.1386	3.3212
	RMSE	1.5931	1.1606	1.1009	0.6633	0.6771	0.6083

this architecture is that it has a memory cell and non-linear gating units which can control the flow of information in and output. Basically, one LSTM cell has three gates (input, forget, and output), block input, a single cell, and an output activation function. Here, the output of the block is feedback to the block input and all gates. It scales to long-term temporal dependencies and solves the exploding and gradient vanishing problem.

2.2.6.5 Deep GRU Regressor A Gated Recurrent Unit (GRU) was developed in 2014 by Cho et al., and each recurrent unit captures different timescale dependencies [58]. It has gating units like the LSTM units that control the flow of information. However, it does not have a separate memory cell.

2.2.6.6 Bidirectional Deep Neural Networks Regressor In sequential tagging tasks, there is a requirement of both past and future input at a given instant t , like past features (via forwarding passes) and future features (via reverse passes)

in each time frame are provided. This can be achieved by the bidirectional neural network. Here, the pass-on unfolded networks in time are done in the same way as a regular network; however, there is a need to unfold the hidden states every time [59].

2.3 Performance Indices

Commonly used performance evaluation indices for evaluating the performance of various regression topologies have been discussed in the literature [60]. These indices evaluate the performance based on the difference between the actual and predicted value at any time t . In this paper, the average error, maximum error, and root mean square are used, and their mathematical expression is presented in Table 1, where P_{ri} and P_{pi} refer to the i_{th} test value and predicted value, respectively.

Table 8 Summary statistics of all models

Category	Mapping	Model	P.I		
			AE	ME	RMSE
Linear	Knee to knee	OMP	3.0343	6.4309	3.5371
	Hip to hip	Lasso	1.5761	3.4735	1.9229
	Ankle to ankle	Lasso	0.9721	2.7256	1.2054
	All to knee	Huber	0.4831	2.1410	0.7368
	All to hip	Huber	0.9979	2.0673	1.1462
	All to ankle	ElasticNet	0.7934	1.6635	0.9078
SVM	Knee to knee	NuSVR	2.9994	6.2751	3.5031
	Hip to hip	NuSVR	1.5197	3.6752	1.9194
	Ankle to ankle	NuSVR	0.9891	2.6223	1.2137
	All to knee	LinearSVR	1.4806	2.9191	1.7311
	All to hip	NuSVR	0.7078	1.9324	0.9614
	All to ankle	NuSVR	0.6027	1.3671	0.6966
KNN	Knee to knee	Weight uniform	3.1085	6.4051	3.7021
	Hip to hip	Weight uniform	1.6648	4.2995	2.1916
	Ankle to ankle	Weight uniform	1.1337	2.8593	1.4132
	All to knee	Weight distance	1.3746	4.3981	1.7829
	All to hip	Weight distance	0.5518	1.3736	0.7121
	All to ankle	Weight distance	0.3291	1.2799	0.4812
Ensemble	Knee to knee	Random forest	3.8664	6.4051	4.6923
	Hip to hip	AdaBoost	1.7429	4.2817	2.1297
	Ankle to ankle	AdaBoost	1.4186	3.2271	1.7044
	All to knee	Gradient Boost	0.8646	2.9906	1.1442
	All to hip	Random forest	0.5096	1.2597	0.6521
	All to ankle	Random forest	0.4513	1.1365	0.5759
Probabilistic	Knee to knee	GPR	3.0438	6.0495	3.5151
	Hip to hip	GPR	1.5494	3.3209	1.8957
	Ankle to ankle	GPR	0.9449	2.3646	1.1596
	All to knee	ARD	0.6541	1.9488	0.8555
	All to hip	GPR	0.8751	1.1231	1.3281
	All to ankle	GPR	0.1685	0.8273	0.2549
Deep learning	Knee to knee	BiGRU	2.5379	6.3704	1.5931
	Hip to hip	BiGRU	1.3471	3.9372	1.1606
	Ankle to ankle	LSTM	0.9155	2.4823	0.9568
	All to knee	BiLSTM	0.3751	2.4749	0.6124
	All to hip	MLP	0.2908	1.0202	0.4409
	All to ankle	LSTM	0.2851	1.6657	0.5339

3 Results and Discussion

In this section, the authors have examined the machine learning techniques discussed in the previous section. Comparative analysis of the developed techniques is based on categories such as linear, support vector machines, k-nearest neighbor, ensemble, probabilistic, and deep learning models. Summary statistic between many-to-one and one-to-one mapping is also discussed in their respective category. The current study has used the MNIT gait dataset. It consists of 120 subjects from the age group of 5–60 years. All subjects have no history of injury and data were collected from walking on a plane. The random state is chosen as 42 for splitting the dataset into training and testing sets for producing the same result for every run time. Symbols A, B, C, D, E, and F are used to represent the knee-to-knee, hip-to-hip, ankle-to-ankle, all-to-knee, all-to-hip, and all-to-ankle mapping, respectively, in summary statistic tables.

Table 2 presents the summary statistics of developed linear models for one-to-one and many-to-one mapping. Results show that the Lasso, OMP, Huber, and ElasticNet regressor perform the best for hip-to-hip, ankle-to-ankle, knee-to-knee, all-to-knee, all-to-hip, and all-to-ankle mapping.

Table 3 allows the comparative analysis of the support vector machines. It shows that NuSVR performs better for all mappings, except all-to-knee mapping where LinearSVR performs the best. In k-nearest neighbors, weight uniform provides a better result for one-to-one, while in many-to-one mapping the weight distance yields the best performance as shown in Table 4.

Table 5 allows the comparison of ensemble models, where the random forest yields better results for all-to-hip, all-to-ankle, and knee-to-knee mapping, whereas the AdaBoost and gradient boost regressor predict better results for both hip-to-hip and ankle-to-ankle mapping and all-to-knee mapping respectively. Overall, the many-to-one mapping performs well because it captures the relationship between the knee, hip, and ankle.

Table 6 presents the summary for the developed probabilistic models. The Gaussian process regressor performs better for all mapping except for all-to-knee, where the ARD yields good results. The performance of GPR is better overall than other methods, because of its non-parametric form, which allows it to capture the uncertainty well.

The performance of neural network-based deep learning methods is discussed in Table 7. Results show that the bidirectional GRU provides a good result for knee-to-knee and hip-to-hip mapping, whereas for ankle-to-ankle mapping LSTM performs best. However, different models perform better for many-one-mapping because of the variation in the knee, ankle, and hip trajectory datasets. Overall,

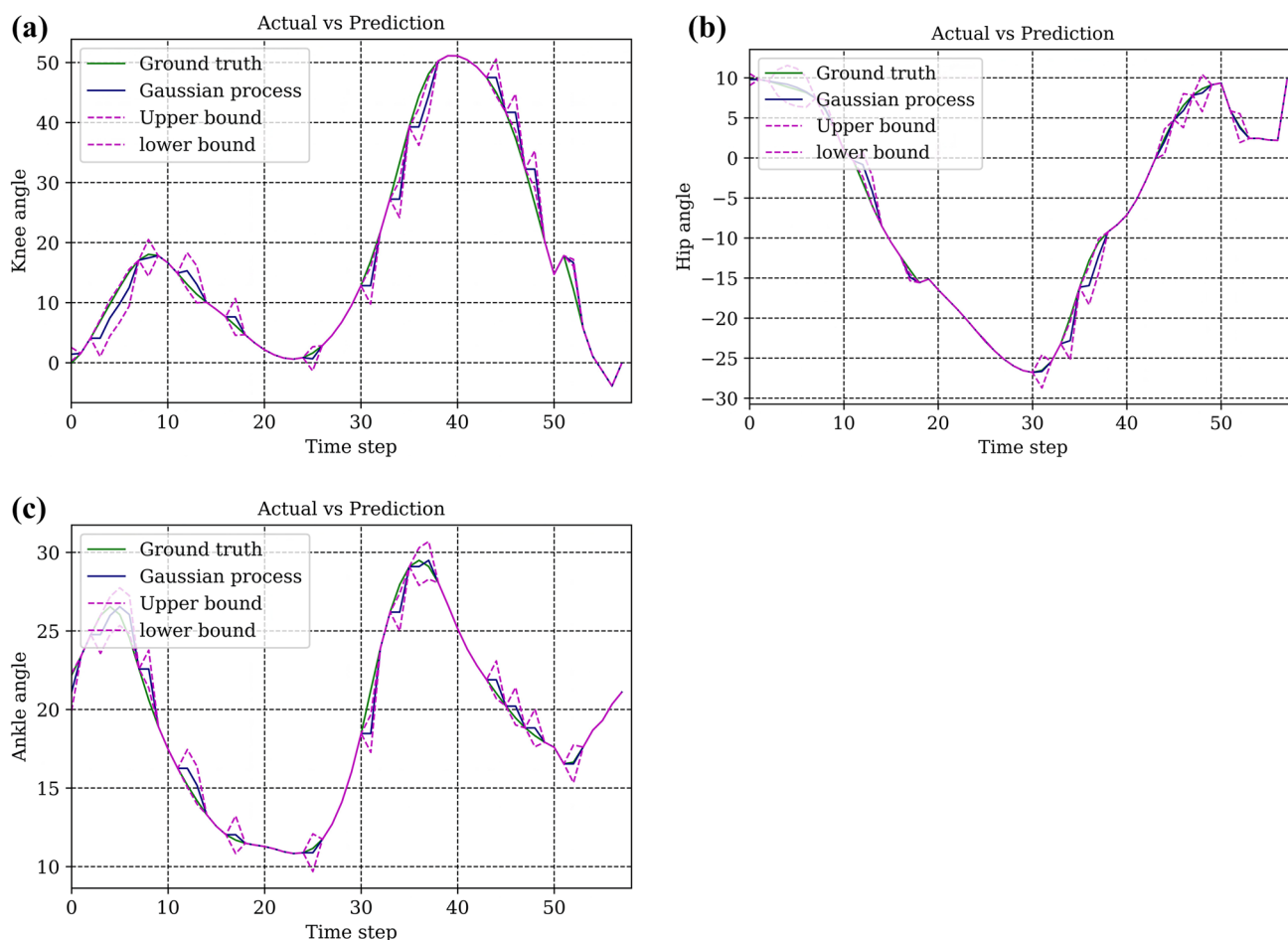


Fig. 3 GP regression for one-to-one mapping. **a** Knee-to-knee mapping. **b** Hip-to-hip mapping. **c** Ankle-to-ankle mapping

many-to-one mapping provides better results than one-to-one mapping for deep learning models.

Table 8 allows the comparative analysis of the developed models in all categories. Comparative analysis between the linear models, support vector machines, k-nearest neighbors, and ensemble learning for different mapping is described as follows: NuSVR performance is best for knee-to-knee and hip-to-hip mapping, and it provides the performance parameter values. However, the Lasso regression provides the minimum M.E. for hip-to-hip mapping, whereas both Lasso and Huber regressor perform well for ankle-to-ankle and all-to-knee mapping, respectively. Likewise, the k-nearest neighbor with weight equal to distance and random forest perform well for all-to-ankle and all-to-hip mapping. So, the different models perform differently in mappings. Therefore, we cannot identify a single best method for all mappings in these categories. However, deep learning models are performed best among all. The maximum error is more than the probabilistic model and the prediction is arbitrary for unseen data, i.e., we cannot be sure about the output prediction with confidence. In the case of our problem, the maximum error

is also an important parameter because in real-time implementation, even a small deviation in trajectory prediction can lead to the system's instability. So, it is necessary to limit this variation. Therefore, to deal with the situation, the uncertainty is included in the model. Thus, the probabilistic model is used, and the performance is best in case of maximum error. In probabilistic models, the GPR model performs well in terms of maximum error and outperforms even deep learning. In terms of other indices also, the GPR performs well.

Figures 3 and 4 show the result of GPR prediction with a 95% confidence interval for one-to-one and many-to-one mappings, respectively. Therefore, it is recommended to use the GPR for trajectory generation models in robotic applications.

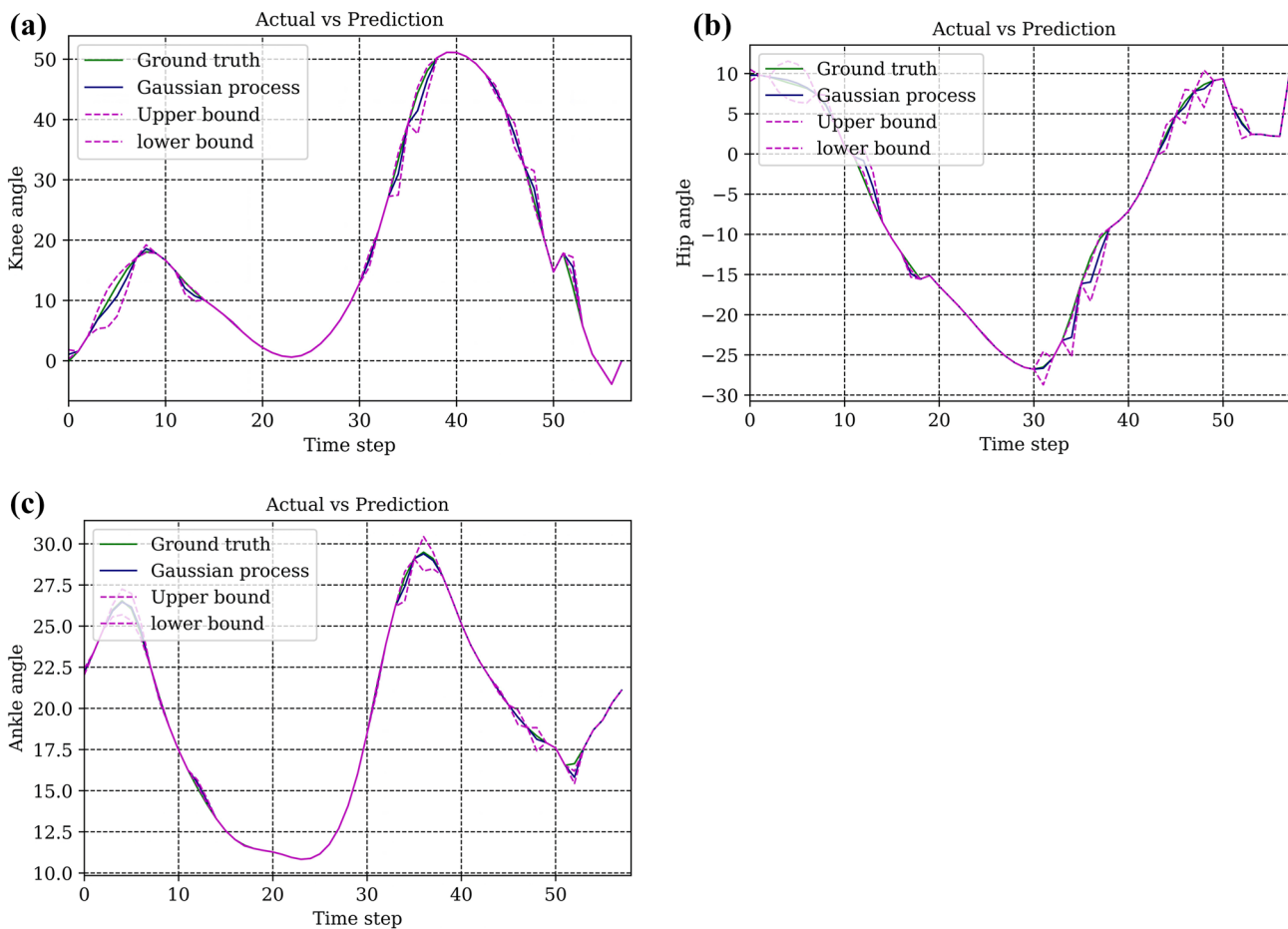


Fig. 4 GP regression for many-to-one mapping. **a** Knee-to-knee mapping. **b** Hip-to-hip mapping. **c** Ankle-to-ankle mapping

4 Conclusion

This paper presented the kinematic modeling of human locomotion using machine learning approaches (linear models, neural networks, support vector machines, nearest neighbor, Gaussian processes, deep learning, several types of regression trees, and ensembles like random forest). The results showed that deep learning methods outperformed the rest of the techniques for one-to-one and many-to-one mapping. In addition, the many-to-one mapping outperforms the one-to-one mapping as well. However, if the problem of model bias becomes evident, then the probabilistic-based Gaussian process regressor shows better utility. The following conclusions can be drawn from this work. Firstly, 32 machine learning models for both mapping (one-to-one mapping and many-to-one mapping) are developed. Secondly, deep learning outperforms all other mapping models. Lastly, it is standard practice that the maximum error should be within the limits, because in real-time implementation of a biped robot, even a small deviation in trajectory prediction can lead to the system

becoming unstable. Therefore, the probabilistic model should be used as the trajectory generation model. Overall, this study can contribute in three ways: (a) provide the reference trajectory generation, (b) next time horizon control input can be predicted and (c) state estimation of joint position. As a future scope, the authors will explore suitable data preprocessing and preprocessing techniques, global optimization of learning models during training, and implementation of cross-validation techniques for a reduction in model bias issues. In addition, an appropriate approximate technique in probabilistic models for handling the long-term predictions will be explored.

Funding None.

Declarations

Conflict of interest All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

Ethics approval and consent to participate Not applicable.

References

- Conde, M. A., Rodriguez-Sedano, F. J., Fernandez-Llamas, C., Goncalves, J., Lima, J., & Garcia-Penalvo, F. J. (2021). Fostering STEAM through challenge-based learning, robotics, and physical devices: A systematic mapping literature review. *Computer Applications in Engineering Education*, 29, 46–65.
- Singh, B., Kumar, R., & Singh, V. P. (2021). Reinforcement learning in robotic applications: a comprehensive survey. *Artificial Intelligence Review*. <https://doi.org/10.1007/s10462-021-09997-9>
- Zheng, T. J., Zhu, Y., Zhang, Z., Zhao, S., Chen, J., & Zhao, J. (2018). Parametric gait online generation of a lower-limb exoskeleton for individuals with paraplegia. *Journal of Bionic Engineering*, 15, 941–949.
- Boudon, B., Linares, J. M., Abourachid, A., Vauquelin, A., & Mermoz, E. (2018). Bio-inspired topological skeleton for the analysis of quadruped kinematic gait. *Journal of Bionic Engineering*, 15, 839–850.
- Gong, Z., Cheng, J., Chen, X., Sun, W., Fang, X., Hu, K., Xie, Z., Wang, T., & Wen, L. (2018). A bio-inspired soft robotic arm: Kinematic modeling and hydrodynamic experiments. *Journal of Bionic Engineering*, 15, 204–219.
- Ma, Y., Fan, X., Cai, J., Tao, J., Gao, Y., & Yang, Q. (2021). Application of sensor data information cognitive computing algorithm in adaptive control of wheeled robot. *IEEE Sensors Journal*. <https://doi.org/10.1109/JSEN.2021.3054058>
- Ficht, G., & Behnke, S. (2021). Bipedal humanoid hardware design: A technology review. *Current Robotics Reports*. <https://doi.org/10.1007/s43154-021-00050-9>
- Liu, B., Xiao, X., & Stone, P. (2021). A lifelong learning approach to mobile robot navigation. *IEEE Robotics and Automation Letters*, 6, 1090–1096.
- Pratt, J., Chew, C. M., Torres, A., Dilworth, P., & Pratt, G. (2001). Virtual model control: An intuitive approach for bipedal locomotion. *The International Journal of Robotics Research*, 20, 129–143.
- Golliday, C., & Hemami, H. (1977). An approach to analyzing biped locomotion dynamics and designing robot locomotion controls. *IEEE Transactions on Automatic Control*, 22, 963–972.
- Park, J. H., Chung, H. (1999) ZMP compensation by online trajectory generation for biped robots. In IEEE SMC'99 Conference Proceedings IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 99CH37028), 4, 960–965.
- Hasegawa, Y., Arakawa, T., & Fukuda, T. (2000). Trajectory generation for biped locomotion robot. *Mechatronics*, 10, 67–89.
- Arakawa, T., Fukuda, T. (1996) Natural motion trajectory generation of biped locomotion robot using genetic algorithm through energy optimization. In 1996 IEEE International Conference on Systems, Man and Cybernetics, Information Intelligence and Systems (Cat. No. 96CH35929), 2, 1495–1500.
- Suzuki, T., Tsuji, T., Shibuya, M., & Ohnishi, K. (2008). ZMP reference trajectory generation for biped robot with inverted pendulum model by using virtual supporting point. *IEEE Transactions on Industry Applications*, 128, 687–693.
- Kim, J., Ba, D. X., Yeom, H., & Bae, J. (2021). Gait optimization of a quadruped robot using evolutionary computation. *Journal of Bionic Engineering*, 18, 306–318.
- Chen, J., Liu, C., Zhao, H., Zhu, Y., & Zhao, J. (2020). Learning to identify footholds from geometric characteristics for a six-legged robot over rugged terrain. *Journal of Bionic Engineering*, 17, 512–522.
- Juang, C. F., & Yeh, Y. T. (2017). Multi-objective evolution of biped robot gaits using advanced continuous ant-colony optimized recurrent neural networks. *IEEE Transactions on Cybernetics*, 48, 1910–1922.
- Embry, K. R., Villarreal, D. J., Macaluso, R. L., & Gregg, R. D. (2018). Modeling the kinematics of human locomotion over continuously varying speeds and inclines. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 26, 2342–2350.
- Clever, D., Hu, Y., & Mombaur, K. (2018). Humanoid gait generation in complex environments based on template models and optimality principles learned from human beings. *The International Journal of Robotics Research*, 37, 1184–1204.
- Riley, M., Ude, A., Wade, K., Atkeson, C. G. (2003) Enabling real-time full-body imitation: A natural way of transferring human movement to humanoids. In 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422), 2, 2368–2374.
- Aithal, C. N., Ishwarya, P., Sneha, S., Yashvardhan, C. N., & Suresh, K. V. (2020). Design of a Bipedal Robot. *Advances in VLSI, Signal Processing, Power Electronics, IoT, Communication and Embedded Systems: Select Proceedings of VSPICE, 2021*, 752–755.
- Prakash, C., Kumar, R., & Mittal, N. (2018). Recent developments in human gait research: Parameters, approaches, applications, machine learning techniques, datasets and challenges. *Artificial Intelligence Review*, 49, 1–40.
- Prakash, C., Gupta, K., Mittal, A., Kumar, R., & Laxmi, V. (2015). Passive marker based optical system for gait kinematics for lower extremity. *Procedia Computer Science*, 45, 176–185.
- Prakash, C., Mittal, A., Kumar, R., Mittal, N. (2015) Identification of spatio-temporal and kinematics parameters for 2-D optical gait analysis system using passive markers. In 2015 International Conference on Advances in Computer Engineering and Applications, 143–149.
- Prakash, C., Mittal, A., Tripathi, S., Kumar, R., Mittal, N. (2016) A framework for human recognition using a multi-model gait analysis approach. In 2016 International Conference on Computing, Communication and Automation (ICCCA), 348–353.
- Prakash, C., Mittal, A., Kumar, R., Mittal, N. (2015) Identification of gait parameters from silhouette images. In 2015 Eighth International Conference on Contemporary Computing (IC3), 190–195.
- Prakash, C., Gupta, K., Kumar, R., Mittal, N. (2016) Fuzzy logic-based gait phase detection using passive markers. In Proceedings of Fifth International Conference on Soft Computing for Problem Solving, Springer, Singapore, 561–572.
- Prakash, C., Sujil, A., Kumar, R., Mittal, N. (2019) Linear prediction model for joint movement of lower extremity. In Recent Findings in Intelligent Computing Techniques, Springer, Singapore, 235–243.
- Prakash, C., Kumar, R., Mittal, N., & Raj, G. (2018). Vision based identification of joint coordinates for marker-less gait analysis. *Procedia Computer Science*, 132, 68–75.
- Weisberg, S. (2005). *Applied linear regression* (p. 528). Wiley.

31. Hoerl, A. E., Kannard, R. W., & Baldwin, K. F. (1975). Ridge regression: Some simulations. *Communications in Statistics-Theory and Methods*, 4, 105–123.
32. Marquardt, D. W., & Snee, R. D. (1975). Ridge regression in practice. *The American Statistician*, 29, 3–20.
33. Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *Annals of Statistics*, 32, 407–499.
34. Then, Q. (2020). *Conversion modeling: Machine learning for marketing decision support* (p. 95). Humboldt University of Berlin.
35. Sun, Q., Zhou, W. X., & Fan, J. (2020). Adaptive huber regression. *Journal of the American Statistical Association*, 115, 254–265.
36. Choi, S., Kim, T., & Yu, W. (1997). Performance evaluation of RANSAC family. *Journal of Computer Vision*, 24, 271–300.
37. Wilcox, R. (1998). A note on the Theil-Sen regression estimator when the regressor is random and the error term is heteroscedastic. *Biometrical Journal: Journal of Mathematical Methods in Biosciences*, 40, 261–268.
38. Zou, H., & Hastie, T. (2005). Regularization, and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67, 301–320.
39. Pati, Y. C., Rezaiifar, R., Krishnaprasad, P. S. (1993) Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In Proceedings of 27th Asilomar Conference on Signals, Systems and Computers, 40–44.
40. Vovk, V. (2013). Kernel ridge regression. In E. Inference (Ed.), *Springer* (pp. 105–116). Germany.
41. Drucker, H., Burges, C. J., Kaufman, L., Smola, A., & Vapnik, V. (1997). Support vector regression machines. *Advances in Neural Information Processing Systems*, 9, 155–161.
42. Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14, 199–222.
43. Cheng, D., Zhang, S., Deng, Z., Zhu, Y., Zong, M. (2014) KNN algorithm with data-driven k value. In International Conference on Advanced Data Mining and Applications, Springer, Cham, 499–512.
44. Chou, P. A. (1991). Optimal partitioning for classification and regression trees. *IEEE Computer Architecture Letters*, 13, 340–354.
45. Breiman, L. (2001). Random forests. *Machine Learning*, 45, 5–32.
46. Schapire, R. E. (2010). The convergence rate of AdaBoost. In *COLT*, 10, 308–309.
47. Zemel, R. S., Pitassi, T. (2001) A gradient-based boosting algorithm for regression problems. In: Advances in Neural Information Processing Systems, 696–702.
48. Raftery, A. E., Madigan, D., & Hoeting, J. A. (1997). Bayesian model averaging for linear regression models. *Journal of the American Statistical Association*, 92, 179–191.
49. Park, T., & Casella, G. (2008). The Bayesian lasso. *Journal of the American Statistical Association*, 103, 681–686.
50. Schulz, E., Speekenbrink, M., & Krause, A. (2018). A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of Mathematical Psychology*, 85, 1–16.
51. Burt, D., Rasmussen, C. E., Van Der Wilk, M. (2019) Rates of convergence for sparse variational Gaussian process regression. In International Conference on Machine Learning, 862–871.
52. Thimm, G., & Fiesler, E. (1997). High-order and multilayer perceptron initialization. *IEEE Transactions on Neural Networks*, 8, 349–359.
53. Murtagh, F. (1991). Multilayer perceptrons for classification and regression. *Neurocomputing*, 2, 183–197.
54. Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast-learning algorithm for deep belief nets. *Neural Computation*, 18, 1527–1554.
55. Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E. (2017). A survey of deep neural network architectures and their applications. *Neurocomputing*, 234, 11–26.
56. Gallicchio, C. (2018) Short-term memory of deep rnn. arXiv preprint <https://arxiv.org/abs/1802.00748>. Accessed 2 Feb 2018.
57. Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2016). LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28, 2222–2232.
58. Chung, J., Gulcehre, C., Cho, K., Bengio, Y. (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint <https://arxiv.org/abs/1412.3555>. Accessed 11 Dec 2014.
59. Huang, Z., Xu, W., Yu, K. (2015) Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint <https://arxiv.org/abs/1508.01991>. Accessed 9 Aug 2015.
60. Sheng, H., Xiao, J., Cheng, Y., Ni, Q., & Wang, S. (2017). Short-term solar power forecasting based on weighted Gaussian process regression. *IEEE Transactions on Industrial Electronics*, 65, 300–308.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.