Prediction of a Ball Trajectory for the Humanoid Robots: A Friction-Based Study

Behnam Yazdankhoo[1], Mohammad Navid Shahsavari[1],
Soroush Sadeghnejad[1](✉), and Jacky Baltes[2]

[1] Bio-Inspired System Design Lab, Amirkabir University of Technology
(Tehran Polytechnic), No. 424, Hafez Avenue, P.O. Box 15875-4413 Tehran, Iran
s.sadeghnejad@aut.ac.ir

[2] Department of Electrical Engineering, National Taiwan Normal University,
162 Heping E Road Sec. 1, Taipei 10610, Taiwan

Abstract. Recent advances in robotics have made it necessary for robots to be able to predict actions like humans. This problem is well presented in international RoboCup competition leagues, especially for humanoid robots in challenges such as Goal-Kick from Moving Ball. In this paper, we proposed double exponential smoothing (DES), autoregressive (AR) and quadratic prediction (QP) as online methods and self-perturbing recursive least squares (SPRLS) as an offline method for prediction of the ball trajectory on ground. These prediction methods are compared in two scenarios by applying LuGre friction model. We simulated our proposed methods by Simmechanics library of MATLAB's Simulink. By comparing results using root-mean-square error and normalized root-mean-square error, we could deduce that methods that were based on predefined models such as QP performed poorly when the friction deviated from the presumed model. Whereas numerical methods such as AR could adapt themselves to variation much better, depending on the friction force variation with time. Also offline methods such as SPRLS are good replacements for online ones when pre-training is possible.

Keywords: Humanoid robots · Ball trajectory prediction · LuGre model · Goal-Kick from Moving Ball · Autoregressive · Exponential smoothing

1 Introduction

The RoboCup competitions goal for 2050 was started in 1997: a team with fully autonomous humanoid soccer player robots shall win against the winner of the world cup [1]. In order to reach this objective, RoboCup competitions are held every year with incremental steps toward this goal [2,3]. One of the technical challenges in RoboCup competitions is Goal-Kick from Moving Ball. Figure 1

B. Yazdankhoo and M. N. Shahsavari—Contributed equally to this work.

D. Holz et al. (Eds.): RoboCup 2018, LNAI 11374, pp. 387–398, 2019.
https://doi.org/10.1007/978-3-030-27544-0_32

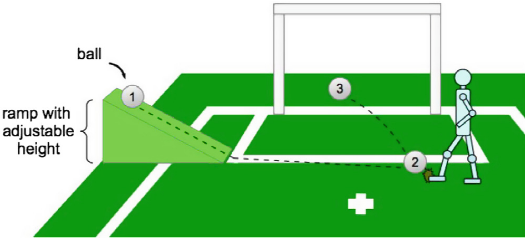

Fig. 1. Steps of the RoboCup Goal-Kick from Moving Ball challenge [4].

shows the steps of this challenge; a ball is thrown and the robot should place itself in the appropriate position and kick the ball.

In order to reach this ability, the robot should be able to predict the ball trajectory. Prediction is one of the primary abilities of the human. A psychology study in 1998 showed that a 6-month-old human infant has the ability to predict the trajectory of the objects [5]. This shows the necessity of the prediction in humanoid robots in order to act and think like humans.

A research by Seekircher *et al.* [6] presented an accurate ball tracking using Extended Kalman Filters in order to implement high level behavior in the RoboCup 3D soccer simulation scenario. There is considerable number of works carried out on tracking problem. However, it is different from trajectory prediction. Wang *et al.* proposed an online intention inference algorithm to predict the intention of the human before hitting the ball in a ping-pong match between human and robot [7]. But no algorithm for prediction of ball trajectory is presented in this work. In [8], a method was presented for catching a thrown ball. They focused on the trajectory of the robot to catch the ball by using a least squares method for prediction of the ball trajectory.

Birbach *et al.* [9] presented a method for estimating position and velocity of multiple flying balls for the purpose of robotic ball catching. To this end, a multi-target recursive Bayes filter, the Gaussian Mixture Probability Hypothesis Density filter (GMPHD), fed by a circle detector was used. Finally, they focused on detections that are likely to lead to a catchable trajectory which increases robustness. However, their work was an estimation of the ball position. A predictor based on nearest neighbor regression was presented by Mironov and Pongratz in [10], which does not require an exact physical model of the motion. The challenge of such application consists of a high volume of calculations that are needed to compare the current trajectory with examples from the database.

A research by Baum *et al.* [11] presented a visual tracking and Extended Kalman Filter based prediction method for catching a flying ball with a Hand-Arm-System. In [12], a method was proposed based on probability hypothesis density (PHD) filtering for predicting the ball trajectory.

These studies were estimating the ball position, either predicting the ball trajectory with methods that have a huge amount of calculation or not focusing on rolling ball on ground with variant values of the friction. In this paper, we focused on predicting the soccer ball trajectory rolling on the ground with different values of the friction compared with the constant value of the friction. Also, we presented different possible scenarios for friction force and demonstrate which method is the best solution for each scenario. We will apply all methods to friction scenarios and demonstrate the best method by using root-mean-square error (RMSE) and normalized root-mean-square error (NRMSE) as comparison factors. The main aim of our work is to predict the trajectory of the rolling ball in order to be kicked towards the goal; however, passing the ball to another robot and using it for a goalkeeper can be other possible applications.

The rest of the paper consists of 4 sections: Sect. 2 presents our proposed prediction methods and formulations of them in details. Section 3 contains information about the friction model that is used in our research and formulation of it. Assumptions and parameters of the simulation along with simulation results are presented in Sect. 4. Finally, in Sect. 5, a summary of our work, conclusions and directions for future works are presented.

2 Prediction Methods

Ball trajectory prediction could be carried out in two different schemes: online and offline. We use three methods for online scheme and one method for offline scheme, although other methods are also available in the literature, for instance recurrent neural networks [13]. The presented methods are valid while the ball is moving on the ground, in any direction; however, without loss of generality, we denote the position by x assuming the ball is moving in X-direction.

2.1 Online Methods

In the online scheme, the trajectory is predicted while the ball is moving, without any a priori knowledge about the environment, i.e. the physical condition of the ground is unknown to the robot.

Double Exponential Smoothing Method (DES). In [14], Exponential Smoothing methods are divided into 15 classes based on trend and seasonality. Ball position, in our case, has clearly no seasonality, but it does have an increasing trend. Therefore, we adopt DES as our prediction method here. In this method, we assume that x is the position of the ball at time t. The predicted positions based on DES is obtained according to the following relations:

$$S_i = \alpha x_i + (1-\alpha)(S_{i-1} + b_{i-1}) \tag{1}$$

$$b_i = \gamma(S_i - S_{i-1}) + (1-\gamma)b_{i-1} \tag{2}$$

where $0 < \alpha, \gamma \leq 1$, and $\hat{x}$ represents predicted position. Also, i $= 1, 2, ..., $n. S_i and b_i are calculated in n steps and are used to derive predicted position, as follows:

$$\hat{x}_{n+m} = S_n + mb_n \tag{3}$$

Where, m $= 1, 2, \ldots$ is the number of steps ahead. When $0 \leq t < T$, Eqs. (1) and (2) are used and the prediction for future positions is during $t \geq T$ using Eq. (3), where T is determined based on the desired conditions.

There are different ways of calculating the coefficients α and γ, for instance in [15] an adaptive approach is described. We used constant values for coefficients in this work, because the defined problem is rather simplified and thus constant coefficients suffice. Nevertheless, adaptive prediction which takes prediction updates into account will be one of the important future works we are pursuing.

Autoregressive Method (AR). An autoregressive process of order p, or AR(p), is one which estimates the future value of a parameter based on a linear combination of p previous values of that parameter. An AR(p) process can be represented according to the following relation,

$$\hat{x}_{i+1} = \phi_1 x_i + \phi_2 x_{i-1} + \ldots + \phi_p x_{i-p+1} \tag{4}$$

where ϕ_j, j $= 1, 2, \ldots,$ p, are the process coefficients and $\hat{x}_{i+1}$ represents the one-step-ahead predicted position.

Some procedures exist in order to determine the appropriate order p; however, it has been shown that second derivative of position, i.e. acceleration, can be represented by a first-order AR model [16]. Therefore, in terms of acceleration, we can write:

$$\hat{a}_{i+1} = \psi a_i \tag{5}$$

where a and $\hat{a}$ represent the actual and predicted accelerations, respectively. Also, the relation between a and x in discrete space can be represented as:

$$a_i = \frac{x_i - 2x_{i-1} + x_{i-2}}{(\delta t)^2} \tag{6}$$

where δt is the corresponding sample time. Combining (5) and (6), a third-order AR model is obtained for position.

$$\hat{x}_{i+1} = (2 + \psi)x_i + (-1 - 2\psi)x_{i-1} + \psi x_{i-2} \tag{7}$$

Coefficients of an AR(p) model can be obtained through different methods. A well-known and efficient method is the Yule-Walker equations [17]. Considering the notation defined in (4), the Yule-Walker equations can be shown in matrix form as follows ((8b) is the equivalent form of (8a)).

$$\begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_p \end{bmatrix} = \begin{bmatrix} r_0 & r_1 & r_2 & \ldots & r_{p-2} & r_{p-1} \\ r_1 & r_0 & r_1 & & r_{p-3} & r_{p-2} \\ & \vdots & & \ddots & \vdots & \\ r_{p-1} & r_{p-2} & r_{p-3} & \cdots & r_1 & r_0 \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_p \end{bmatrix} \tag{8a}$$

$$\boldsymbol{r} = \boldsymbol{R\Phi} \tag{8b}$$

In (8a), r_k, k $= 0, 1, \ldots,$ p, are the autocorrelation functions which are defined according to:

$$r_k := \frac{c_k}{c_0} \tag{9}$$

where c_k are the autocovariance functions which are defined as follows:

$$c_k := E[x_i x_{i-k}] \tag{10}$$

where $E[.]$ denotes mathematical expectation. Theoretically, $E[x_i x_{i-k}]$ should be computed with infinite number of observations. However, since this is not possible during an online prediction, the mathematical expectation could be approximated by proper number of observations (based on the parameter T explained in the previous section). Also note that for autocovariance function, we have $c_{-k} = c_k$.

Using (9) and (10), the corresponding matrices $\mathbf{r}$ and $\mathbf{R}$ in (8a and 8b) can be obtained and, thus, the matrix of coefficients $\boldsymbol{\phi}$ is achieved.

$$\boldsymbol{\Phi} = \boldsymbol{R}^{-1}\boldsymbol{r} \tag{11}$$

For the acceleration model represented by (5), the single coefficient ψ can be obtained by using Eqs. (8a and 8b) to (11). Then, substituting the resulted ψ into (7), the third order model for one-step-ahead position is attained. To predict position m steps ahead, one should repeat the Eq. (7) m times. For instance, for x_{i+m} we can write:

$$\hat{x}_{i+m} = (2+\psi)\hat{x}_{i+m-1} + (-1-2\psi)\hat{x}_{i+m-2} + \psi\hat{x}_{i+m-3} \tag{12}$$

where $\hat{x}_{i+m-1}$ is the predicted position for m-1 steps ahead, and $\hat{x}_{i+m-2}$ and $\hat{x}_{i+m-3}$ are also defined in a similar manner.

Quadratic Prediction Method (QP). If we consider that the friction is the only force acting on the ball and it follows Coulomb's law, a simplified kinetic model is attained utilizing Newton's second law.

$$a = \frac{\sum F}{m_b} = \frac{-\mu m_b g}{m_b} = -\mu g \tag{13}$$

where a and m_b are the acceleration and mass of the ball, respectively, μ is the coefficient of kinetic friction, which is considered constant, $\sum F$ represents the sum of the forces acting on the ball, and g is the gravitational acceleration.

The kinematic equation representing this motion is:

$$x = \frac{1}{2}at^2 + v_0 t + x_0 \tag{14}$$

where v_0 and x_0 denote the initial velocity and position of the ball, respectively, and t represents the time.

According to (13), the acceleration of the ball is constant and, thus, the position of the ball, defined by (14), represents a quadratic model in terms of time. Therefore, we can generally write:

$$x = A_1 t^2 + A_2 t + A_3 \tag{15}$$

where A_1, A_2 and A_3 are constants. If we can record three different positions at three different times at the beginning T seconds of the motion, coefficients A_1, A_2 and A_3 can be readily used in order to form the equation of motion of the ball according to (15). Having the equation of motion, we can predict the position of the ball at each time during the motion for $t \geq T$.

2.2 Offline Methods

In offline scheme the ball movement is repeated several times and the robot records the data received from the movements before the desired movement begins and learns how to predict the ball trajectory next time that the ball moves.

Self-Perturbing Recursive Least Squares (SPRLS). RLS is a recursive algorithm for determining the parameters of the system. Because of high error of RLS in low variance, we choose SPRLS [18] for predicting the ball position.

$$L_i = \frac{P_{i-1}\Phi_i}{1 + \Phi_i^T P_{i-1}\Phi_i} \tag{16}$$

$$P_i = P_{i-1}(I - L_i \Phi_i^T) + \beta NINT(\lambda e_{i-1}^2) I \tag{17}$$

$$\hat{\theta}_i = \hat{\theta}_{i-1} + L_i(y_i - \Phi_i^T \hat{\theta}_{i-1}) \tag{18}$$

where $e := y - \hat{y}$ is the estimation error with $y := \theta^T \Phi$ and $\hat{y} := \hat{\theta}^T \Phi$ denoting real and estimated outputs, respectively. Also Φ is vector of input parameters, θ is vector of estimated parameters, I is identity matrix, β is design coefficient, λ is sensitivity coefficient, and NINT(.) function is defined as:

$$NINT(x) := \begin{cases} x & x \geq 0.5 \\ 0 & 0 \leq x < 0.5 \end{cases} \tag{19}$$

If we consider that the acceleration is constant, by combination of (13) and (14) we have:

$$x = -\frac{1}{2}\mu g t^2 + v_0 t + x_0 \tag{20}$$

Hence, we can write

$$-2(x - v_0 t - x_0) = \mu g t^2 \tag{21}$$

Comparison between (21) and $y := \theta^T \Phi$ results in:

$$y = -2(x - v_0 t - x_0) \tag{22}$$

$$\theta^T = \mu g \tag{23}$$

$$\phi = t^2 \tag{24}$$

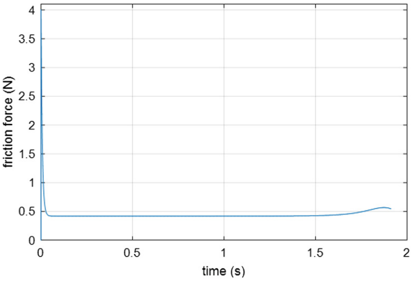

Fig. 2. First scenario friction force-time diagram.

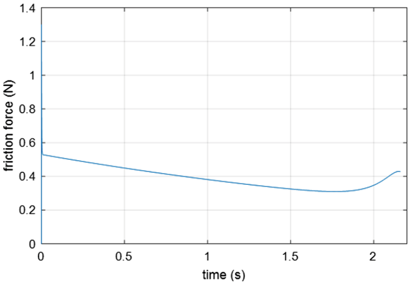

Fig. 3. Second scenario friction force-time diagram.

3 Friction Model

There are several models of friction to use in simulation. Here we have used LuGre model [19] because of simplicity and also good accuracy in many cases. The formulation of this model is:

$$\begin{aligned} F_f &= \sigma_0 z + \sigma_1 \dot{z} + \sigma_2 v \\ \dot{z} &= v - \frac{\sigma_0 |v|}{s(v)} z \\ s(v) &= F_c + (F_s - F_c) exp(-(\frac{v}{v_s})^{\delta_{vs}}) \end{aligned} \tag{25}$$

where F_f is the friction force, σ_1 and σ_2 are damping coefficients that are related to the presliding and kinetic friction states, respectively, v is the velocity, the parameter v_s determines how quickly $s(v)$ approaches F_c, $F_s = \mu_s m_b g$ and $F_c = \mu_k m_b g$ are static friction force and coulomb friction force, respectively, μ_s and μ_k denote friction coefficients and δ_{vs} is the shape factor of Stribeck curve.

We assumed two scenarios with different friction forces in order to investigate the effect of friction on prediction quality, although many other possible scenarios exist. As shown in Fig. 2, the friction in first scenario is almost constant over time. Whereas Fig. 3 shows that it is variant over time in the second scenario. The common coefficients between two scenarios are (the value for δ_{vs} is derived from [20], while the others are chosen arbitrarily in order to obtain reasonable friction forces. Also, the mass of the ball is the standard RoboCup ball mass): $m_b = 0.425\,\text{kg}, v_s = 0.1\,\frac{\text{m}}{\text{s}}, \delta_{vs} = 1$.

Scenario 1 coefficients are: $\mu_s = 0.25, \mu_k = 0.1, \sigma_0 = 30\,\frac{\text{N}}{\text{m}}, \sigma_1 = 2\,\frac{\text{N s}}{\text{m}}, \sigma_2 = 0\,\frac{\text{N s}}{\text{m}}$ and scenario 2 coefficients are: $\mu_s = 0.15, \mu_k = 0.06, \sigma_0 = 100\,\frac{\text{N}}{\text{m}}, \sigma_1 = 0.5\,\frac{\text{N s}}{\text{m}}, \sigma_2 = 0.14\,\frac{\text{N s}}{\text{m}}$.

4 Simulation

4.1 Assumptions and Parameters

In our simulation, we assumed $T = 0.5\,\mathrm{s}$, initial velocity, $v_0 = 2\,\frac{\mathrm{m}}{\mathrm{s}}$ and in order to compare our prediction methods, we use RMSE and NRMSE which start from $t = T$.

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(x_i - \hat{x}_i)^2} \tag{26}$$

$$NRMSE = \frac{RMSE}{x_{\max} - x_{\min}} \tag{27}$$

x and $\hat{x}$ are the actual and predicted positions, respectively, and M is the total number of predicted steps. Also $x_{\max} = x(t = t_{end})$ where t_{end} denotes the time at which the ball stops, and $x_{\min} = x(t = 0)$ for offline method and $x_{\min} = x(t = T)$ for online methods.

We assumed DES method parameters as: $\alpha = 0.9, \gamma = 0.5$

For QP method, in order to derive A_1, A_2, A_3 we need three equations. After T seconds, using positions at $t_1 = 0$, $t_2 = \frac{T}{2}$ and $t_3 = T$ and (15) we have three equations to derive A_1, A_2, A_3 then we have:

$$\begin{aligned} A_1 &= \frac{(t_3x_0 - t_3x_1 - t_2x_0 + t_2x_2)}{t_3t_2(t_3 - t_2)} \\ A_2 &= \frac{-(t_3^2x_0 - t_3^2x_1 - t_2^2x_0 + t_2^2x_2)}{t_3t_2(t_3 - t_2)} \\ A_3 &= x_0 \end{aligned} \tag{28}$$

In SPRLS method, we assumed $v_0 = 2.5\,\frac{\mathrm{m}}{\mathrm{s}}$ for training phase and $v_0 = 2\,\frac{\mathrm{m}}{\mathrm{s}}$ for prediction phase. Also, RMSE and NRMSE are calculated from $t = 0$. Initial values for SPRLS method are: $\hat{\theta}_0 = 1.962, P_0 = 1, e_0 = 1$ and the parameters are: $\lambda = 100, \beta = 30000$.

4.2 Simulation Results

We simulated our methods by means of Simmechanics library (second generation) of MATLAB's Simulink. Our vision system frequency is 20 Hz and in order to achieve more real results, we chose this frequency for running the simulation in MATLAB.

As can be seen in Figs. 4 and 5, the QP method has yielded by far the best result for scenario 1, where the friction force is almost constant over time. This fact can be further proved quantitatively by Tables 1 and 2, where the RMSE and NRMSE are obtained to be 0.0036 m and 0.0034, respectively, for this method. This is, however, not surprising since the predefined model in QP method is in great accordance with the friction force which is applied to the ball during the whole motion in this scenario. Apart from this, the offline SPRLS method has resulted in a good prediction with an acceptable NRMSE of 0.1913. The other two online methods, namely AR and DES, showed poorer predictions with NRMSE of 38.68% and 41.52%, respectively.

For scenario 2, Table 2 indicates that online QP and offline SPRLS still lead to more accurate predictions. However, from another perspective, the 4239% rise in the RMSE for QP method from scenario 1 to scenario 2 shows that when the friction force does not comply with the predefined model in QP, the precision of this method decreases drastically. Meanwhile, the 45.6% increase in the RMSE of SPRLS method shows that such a remarkable change has not occurred for this method. The interesting point in scenario 2 is, however, the 30% reduction of the RMSE in AR method in comparison with scenario 1, which demonstrates that this method has adapted itself better to scenario 2. But this is not the case for DES since its performance has deteriorated in scenario 2.

Looking more closely at Figs. 5 and 7, one can obviously observe that both AR and DES methods have predicted a linear trajectory for the ball, which means that the ball will move infinitely and will never stop. The same concept can be interpreted by investigating the relevant mathematical relations presented in Sect. 2.1. This is, however, not compatible with the physical reality of the problem. Therefore, it can be deduced that methods which merely rely on numerical data for prediction often fail to take the physical concepts into account, and are thus suitable for the situations where little data is available about the actual circumstances beforehand. These methods can also be utilized when the prediction is to be carried out not far into the future; linear predictions can be a good approximation of the curved ball position graph (with respect to time) dependent on the applied friction force. Of course, great care should be taken in adopting numerical methods. As discussed in the previous paragraph, AR method yielded a much better result in scenario 2 (the friction force of which was depicted in Fig. 3) than DES method. However, for a different form of friction, the same result is not guaranteed to be attained, which shows the importance of the physical conditions governing the problem.

Ultimately, it can be concluded that when the physical conditions of the ball and ground are known to a good extent beforehand, model-based methods such as online QP are suitable choices for predicting the trajectory of the ball. If, on the other hand, online predicting is not required and the opportunity for pre-training exists, offline SPRLS method provides a good chance to estimate the physical conditions governing the problem and thus to predict the trajectory by an acceptable accuracy. Moreover, online model-free or numerical methods such as AR or DES can also provide us with good predictions, but the precision of the results greatly depend on the variation of the friction force with time (Fig. 6).

Table 1. Calculated RMSE for all methods

	Scenario1 (m)	Scenario2 (m)
DES	0.4405	0.5669
AR	0.4104	0.3154
QP	0.0036	0.1562
SPRLS	0.3650	0.5316

Table 2. Calculated NRMSE for all methods

	Scenario 1	Scenario 2
DES	0.4152	0.5032
AR	0.386	0.2800
QP	0.0034	0.1386
SPRLS	0.1913	0.2688

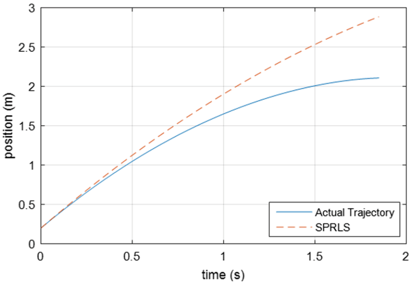

Fig. 4. Trajectory predicted using offline method comparing with actual trajectory in first scenario.

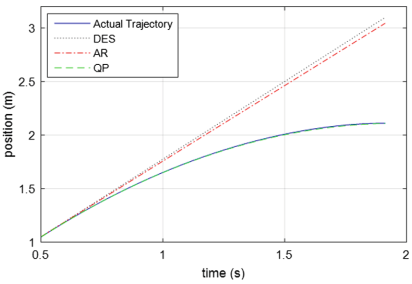

Fig. 5. Trajectory predicted using online methods comparing with actual trajectory in first scenario.

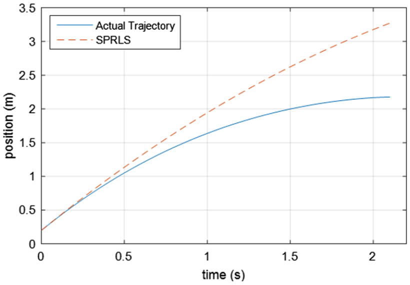

Fig. 6. Trajectory predicted using offline method comparing with actual trajectory in second scenario.

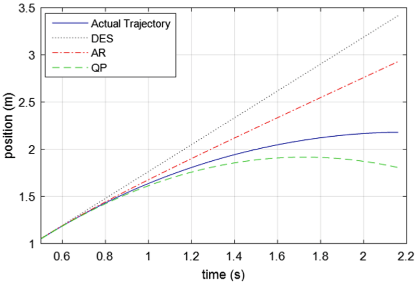

Fig. 7. Trajectory predicted using online methods comparing with actual trajectory in second scenario.

5 Conclusion and Future Works

In this paper, we aimed at predicting the trajectory of the ball for humanoid robots, which can be used for different goals such as the Goal-Kick from Moving Ball challenge in Robocup. We simulated two different friction forces and compared three online and one offline methods for this aim. The simulated scenarios were presented in one direction; however, the equations are all valid while the motion is completely on the ground.

The results of the simulations indicated that the accuracy of the prediction methods is highly dependent on the frictional condition which governs the motion. However, we concluded that if the physical conditions are known before the start of the motion, online model-based methods such QP are good candidates, while model-free AR and DES methods could be utilized online when either little information about motion is available or the time interval of the prediction is rather small. Offline SPRLS method is also useful when the robot has the opportunity to be pre-trained.

There are many directions regarding future works. First, these methods should be implemented on real humanoid robots to predict the ball trajectory in order to kick, pass or intercept the ball in real-life scenarios, since the explored friction force scenarios in this paper were only two examples of various possible ones. Predicting the three-dimensional trajectory of the ball is also of great interest. As another future direction, more realistic and human-like factors can be taken into account. For instance, the prediction could be updated and improved as the ball moves nearer to the robot by considering proper adaption laws for the presented methods. Also other factors which lead to deviation from ideal circumstances should be considered, such as small obstacles along the path of the ball, lack of perfect roundness of the ball and air drag.

References

1. Gerndt, R., Seifert, D., Baltes, J.H., Sadeghnejad, S., Behnke, S.: Humanoid robots in soccer: robots versus humans in RoboCup 2050. IEEE Robot. Autom. Mag. **22**, 147–154 (2015). https://doi.org/10.1109/MRA.2015.2448811
2. Baltes, J., Sadeghnejad, S., Seifert, D., Behnke, S.: RoboCup humanoid league rule developments 2002–2014 and future perspectives. In: Bianchi, R.A.C., Akin, H.L., Ramamoorthy, S., Sugiura, K. (eds.) RoboCup 2014. LNCS (LNAI), vol. 8992, pp. 649–660. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18615-3_53
3. Javadi, M., Azar, S.M., Azami, S., Ghidary, S.S., Sadeghnejad, S., Baltes, J.: Humanoid robot detection using deep learning: a speed-accuracy tradeoff. In: Akiyama, H., Obst, O., Sammut, C., Tonidandel, F. (eds.) RoboCup 2017. LNCS (LNAI), vol. 11175, pp. 338–349. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00308-1_28
4. RoboCup Soccer Humanoid League Laws of the Game (2017). Robocup Humanoid League. http://www.robocuphumanoid.org/wp-content/uploads/RCHL-2017-final-2.pdf
5. von Hofsten, C., Vishton, P., Spelke, E.S., Feng, Q., Rosander, K.: Predictive action in infancy: tracking and reaching for moving objects. Cognition **67**, 255–285 (1998). https://doi.org/10.1016/S0010-0277(98)00029-8
6. Seekircher, A., Abeyruwan, S., Visser, U.: Accurate ball tracking with extended kalman filters as a prerequisite for a high-level behavior with reinforcement learning. In: The 6th Workshop on Humanoid Soccer Robots at Humanoid Conference, Bled, Slovenia (2011)
7. Wang, Z., et al.: Probabilistic movement modeling for intention inference in human-robot interaction. Int. J. Robot. Res. **32**, 841–858 (2013). https://doi.org/10.1177/0278364913478447

8. Nishiwaki, K., Ionno, A., Nagashima, K., Inaba, M., Inoue, H.: The humanoid Saika that catches a thrown ball. In: 6th IEEE International Workshop on Robot and Human Communication, Sendai, Japan, pp. 94–99. IEEE (1997). https://doi.org/10.1109/ROMAN.1997.646959
9. Birbach, O., Frese, U.: Estimation and prediction of multiple flying balls using probability hypothesis density filtering. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3426–3433. IEEE (2011). https://doi.org/10.1109/IROS.2011.6094622
10. Mironov, K., Pongratz, M.: Fast kNN-based prediction for the trajectory of a thrown body. In: 24th Mediterranean Conference on Control and Automation (MED), pp. 512–517. IEEE (2016). https://doi.org/10.1109/MED.2016.7536007
11. Bäuml, B., Wimböck, T., Hirzinger, G.: Kinematically optimal catching a flying ball with a hand-arm-system. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, pp. 2592–2599. IEEE (2010). https://doi.org/10.1109/IROS.2010.5651175
12. Birbach, O.: Tracking and calibration for a ball catching humanoid robot. Doctoral thesis, Universität Bremen (2012)
13. Stolzenburg, F., Michael, O., Obst, O.: Predictive neural networks. arXiv preprint arXiv:180203308 (2018)
14. Gardner Jr., E.S.: Exponential smoothing: the state of the art-part II. Int. J. Forecast. **22**, 637–666 (2006). https://doi.org/10.1016/j.ijforecast.2006.03.005
15. Stakem, F., AlRegib, G.: An adaptive approach to exponential smoothing for CVE state prediction. In: 2nd International Conference on Immersive Telecommunications, Berkley, USA, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), pp. 141–146 (2009). https://doi.org/10.4108/ICST.IMMERSCOM2009.6409
16. Sakr, N., Georganas, N.D., Zhao, J., Shen, X.: Motion and force prediction in haptic media. In: 2007 IEEE International Conference on Multimedia and Expo, Beijing, China, pp. 2242–2245. IEEE (2007). https://doi.org/10.1109/ICME.2007.4285132
17. Cheng, B.: Yule-walker equations. Wiley StatsRef: Statistics Reference Online (2014). https://doi.org/10.1002/9781118445112.stat05549
18. Park, D.-J., Jun, B.-E.: Selfperturbing recursive least squares algorithm with fast tracking capability. Electron. Lett. **28**, 558–559 (1992). https://doi.org/10.1049/el:19920352
19. Piatkowski, T.: Dahl and LuGre dynamic friction models-the analysis of selected properties. Mech. Mach. Theory **73**, 91–100 (2014). https://doi.org/10.1016/j.mechmachtheory.2013.10.009
20. Johanastrom, K., Canudas-De-Wit, C.: Revisiting the LuGre friction model. IEEE Control Syst. **28**, 101–114 (2008). https://doi.org/10.1109/MCS.2008.929425