How to run a world record? A Reinforcement Learning approach

*Sajad Shahsavari, Eero Immonen
Computational Engineering and Analysis (COMEA)
Turku University of Applied Sciences
20520 Turku, Finland
Email: *sajad.shahsavari@turkuamk.fi

Masoomeh Karami, Hashem Haghbayan,
and Juha Plosila
Department of Computing
University of Turku (UTU)
20500 Turku, Finland

KEYWORDS

Optimal Control, Optimal Pacing Profile, Reinforcement Learning, Competitive Running Race

ABSTRACT

Finding the optimal distribution of exerted effort by an athlete in competitive sports has been widely investigated in the fields of sport science, applied mathematics and optimal control. In this article, we propose a reinforcement learning-based solution to the optimal control problem in the running race application. Well-known mathematical model of Keller is used for numerically simulating the dynamics in runner's energy storage and motion. A feed-forward neural network is employed as the probabilistic controller model in continuous action space which transforms the current state (position, velocity and available energy) of the runner to the predicted optimal propulsive force that the runner should apply in the next time step. A logarithmic barrier reward function is designed to evaluate performance of simulated races as a continuous smooth function of runner's position and time. The neural network parameters, then, are identified by maximizing the expected reward using on-policy actor-critic policy-gradient RL algorithm. We trained the controller model for three race lengths: 400, 1500 and 10000 meters and found the force and velocity profiles that produce a near-optimal solution for the runner's problem. Results conform with Keller's theoretical findings with relative percent error of 0.59% and are comparable to real world records with relative percent error of 2.38%, while the same error for Keller's findings is 2.82%.

I INTRODUCTION

I-A Background and motivation

A competitive runner attempts to optimize their *pacing* to minimize the time spent in covering a given distance. This optimal pacing (or velocity) profile is always a trade-off between moving faster and saving energy during the race, and is specific to the individual's physique characteristics. The optimal pacing problem has been widely studied in sports science (e.g. Abbiss and Laursen (2008); Casado, Hanley, Jiménez-Reyes, and Renfree (2021); Jones, Vanhatalo, Burnley, Morton, and Poole (2010)) as well as mathematical modeling (e.g. Alvarez-Ramirez (2002)) and optimal control theory (e.g. Keller (1974); Reardon (2013); Woodside (1991)). Through technological advances in high-performance computing as well as in wearable devices, which today are capable of predicting the running power on-line, the problem has also been addressed by computational optimization (e.g. Aftalion (2017); Maroński and Rogowski (2011)). Finding the optimal running power profile that yields minimum race time is carried out by solving a dynamical optimization problem, subject to constraints from human metabolism and race conditions. Although theoretical solutions for this optimal control problem are proposed in prior research, they fail on more complicated, higher order mathematical models and constraints. On the other hand, numerical approaches, such as nonlinear programming in the so-called direct methods, suffer from *local convergence*, i.e., tending to converge to solutions close to the supplied first guesses, thus, requiring a good initial guess for the solution. This article addresses these issues by using probabilistic controller model trained by reinforcement learning (RL).

Optimizing the pacing strategy for a given race is a well-known open problem (Aftalion & Trélat, 2021). This problem is non-trivial because of: (1) the complexity of the human body metabolism (processes of aerobic and anaerobic energy production/consumption), (2) individual variations in the human physique, (3) mathematical modeling of human metabolism considering individual variations, and (4) resolving the corresponding optimal control problem based on the developed mathematical model.

The study of mathematical models of human body metabolism tries to specify the differential equations that describe the energy (or oxygen) variation in one's body based on the recovery rate and amount of applied work. Typically these models include a number of physiological parameters that characterize the athlete's body and need to be identified, individually, based on the experimental data on athlete's performance. Among these parameters, oxygen uptake rate ($V\dot{O}2$, rate of oxygen recovery or energy production during the exercise) and critical power (CP, highest possible long-lasting rate of energy consumption) are the most significant ones. Beside running races, mathematical modeling for kinetic and metabolic variability has been used in a wide variety of other competitive sports such as road cycling (Wolf, 2019), swimming (McGibbon, Pyne, Shephard, & Thompson, 2018), horse racing (Mercier & Aftalion, 2020), wheelchair athletics (Cooper, 1990), etc.

The challenge of resolving optimal control profile is not restricted in the sport sciences. Generally, optimal control of dynamical systems governed by differential equations, as a fundamental problem in mathematical optimization, has numerous applications in scientific and engineering research. Such practical applications, for example, include space shuffle reentry trajectory optimization (Rahimi, Dev Kumar, & Alighanbari, 2013), unemployment minimization in economics and management (Kamien & Schwartz, 2012), robotic resource manage-

Communications of the ECMS, Volume 36, Issue 1,
Proceedings, ©ECMS Ibrahim A. Hameed, Agus Hasan,
Saleh Abdel-Afou Alaliyat (Editors) 2022
ISBN: 978-3-937436-77-7/978-3-937436-76-0(CD) ISSN 2522-2414

ment, task allocation (Elamvazhuthi & Berman, 2015) and etc., all of which are ultimately reduced to the problem of finding a sequence of decisions (control variables) over a period of time which optimizes an objective function. Analytical solutions such as Linear–Quadratic Regulator (LQR) can solve optimal control problem for linear systems with quadratic cost function, but usually systems are nonlinear and state-constrained. Numerical solutions for such nonlinear problems can be categorized into two classes: (1) indirect methods: apply first-order necessary optimality conditions based on Pontyagin's Maximum Principle to turn the optimal control problem into a multi-point boundary value problem, which can be solved by an appropriate solver, and (2) direct methods: discretize the problem on time, approximate control input, state, or both with parametric functions and iteratively identify the best parameter set which optimizes the objective function (Böhme & Frank, 2017).

In the current article, we propose a solution based on reinforcement learning to the optimal control problem in running races. In essence, the approach is to use a statistical parameterized control model driven towards the optimal solution by experiencing in the simulated race, with no *a priori* assumptions, whilst sufficiently generic to be applied on a variety types of dynamical models. Moreover, unlike the widely used direct methods in optimal control, here we optimize the parameterized *probabilistic policy function*, enabling it to explore the control trajectory space and iteratively improve its performance. We utilize the well-known mathematical model of Keller (1973) representing the dynamics of motion and human body's energy conservation in the running race application. We used theoretical optimal solution for this model provided by Keller (1974) to validate the solution of our proposed method, since physiological constants of a world-champion runner are also identified in Keller (1973), enabling us to reproduce the exact same dynamical model. It is interesting that while Keller assumes a 3-stage run (initial, middle, end), here this 3-part profile is not assumed *a priori*, but it is part of the solution.

I-B Contributions and key limitations

Overall, this paper presents a machine learning solution that is able to learn the optimal pacing profiles predicted by Keller's theory of competitive running. More specifically, the main contributions of the present work include:

- A neural network-based probabilistic policy model transforming the state of the system into optimal control action in continuous-space.
- A reinforcement learning-based parameter optimization procedure to iteratively improve the policy model's performance.
- Numerical simulation of the dynamical system that is essentially an implementation of Keller's mathematical model (differential equations governing runner's force, energy and velocity dynamics) temporally discretized by the Runge–Kutta approximation method (Tan & Chen, 2012).
- Validation discussion on the proposed method by comparing the predictions of our model with the theoretical results provided by Keller (1973).

It is important to note that the study of convergence of the learner and sensitivity of the approximation (time-discretization of the numerical model due to change in the step size) is performed qualitatively and not studied thoroughly in the present research. Another limitation of the current work is that the proposed RL procedure requires many number of simulated experiences in the environment. Nonetheless, the optimized model by simulated experience could potentially be used as a reasonable initial point of an on-line real-world system, thus reducing the number of required training data samples. Besides, even though the proposed analytical techniques in the literature tries to consider several characteristics/features of the runners' physical characteristics and the environment at the same time, the complex nature of modeling an individual runner makes the solution not to be globally suitable for all the runners. Different physical characteristics of the runners and different environments wherein the runners act provide a wide range of features that affect the constant parameters used in the models and/or might even change the mathematical modeling. Another fact is that the mental and psychological features that might significantly affect the modeling are mainly the result of runner and its environment interaction. In this paper, instead of a bottom-up structural/formal solution for the runner's optimal energy consumption, we propose a *high-level behavioural* model that can be trained and refined over time and can consider and accept many features all in one model. Moreover, the proposed computational approach herein is easily portable to different mathematical models, sports, terrains and even in different applications altogether, like battery electric vehicles.

I-C Relation to previous work

This work is directly linked to Keller (1973). This model is a pioneering mathematical model relying on Newton's law of motion and an energy conservation model assuming that the $\dot{VO}2$ is constant during the race (the model is fully described in detail in Subsection II-A). Several modifications to this fundamental work have been introduced by considering: the effect of fatigue (Woodside, 1991), wind resistance and altitude (Quinn, 2004), variation in oxygen uptake rate (Behncke, 1997), and etc. To incorporate aerobic and anaerobic energy, a conceptual hydraulic model has been developed (Morton, 2006) comprised of two energy tanks: storage of aerobic energy which has infinite capacity but limited consumption rate, and storage of anaerobic energy which has unlimited consumption rate but is limited in capacity. Aftalion and Bonnans (2014) has utilized the dynamics of this hydraulic model and solved the optimal running problem for 1500-meter race by Runge-Kutta discretization scheme and nonlinear programming solver, all implemented in the Bocop toolbox (Bonnans, Martinon, & Grélard, 2012).

Reinforcement learning has been used for optimal control of dynamical systems in different applications (see for example J. Duan et al. (2019) for optimal charge/discharge control of hybrid AC-DC microgrids or Liu, Xie, and Modiano (2019) on computer network

queueing systems), but to the authors' knowledge it has not been used in optimal control of competitive sports.

I-D Organization of the article

Next sections of this paper is structured as follows: In section II, we formally define the runner's problem and details of the proposed reinforcement learning-based solution. Afterward, validation experiments are described and predictions of the control input for three track lengths are presented in section III. Finally, we conclude the paper in section IV and discuss future directions.

II PROPOSED METHOD

II-A Formal case study definition

Keller (1973) considered a mathematical model to incorporate the applied force dynamics and runner's energy conservation. The dynamical model is described shortly here for clarification.

The time T to run the track with length D is related to the velocity profile $v(t)$ by:

$$D = \int_0^T v(t)\, dt \tag{1}$$

Governing differential equation of runner's force balance is defined as:

$$\frac{dv(t)}{dt} + \frac{v(t)}{\tau} = f(t) \tag{2}$$

with $v(t)$ being the instantaneous runner's velocity, $f(t)$ the total propulsive force per unit mass applied by the runner and τ the damping/friction constant coefficient. Note that propulsive force $f(t)$ is under control of the runner through which the velocity is manipulated. The initial condition is $v(0) = 0$, i.e., initially the runner is at rest, and the upper bound limit for propulsive force is

$$f(t) \leq F_{max}, \forall t \in [0, T] \tag{3}$$

Dynamics of the runner's energy storage is described by differential equation:

$$\frac{dE(t)}{dt} = \sigma - f(t) \cdot v(t) \tag{4}$$

in which $E(t)$ is the quantity of available muscular energy per unit mass at given time, σ is the rate of energy recovery per unit mass at which energy is supplied to the body by the respiratory and circulatory systems (in excess of the non-running metabolism). Note that initially the runner has a certain amount of available energy in their muscles, i.e., $E(0) = E_0$, and energy can never be negative, i.e.,

$$E(t) \geq 0, \forall t \in [0, T] \tag{5}$$

The optimal control problem is to find $f(t)$, $v(t)$ and $E(t)$ such that (2)-(5) and initial conditions are satisfied and T defined by (1) is minimized. The four physiological constants τ, F_{max}, σ and E_0 and the distance D are given.

Keller (1974) attained analytical solution to this problem by using calculus of variation with optimization of constrained differential equations. He determined the optimal velocity profile under three assumptions: (1) the race should be divided into exactly three distinct phases, (2) the runner should start with maximum force in the first phase ($f(t) = F_{max}$, for $t \in [0, t_1]$), and (3) the runner should deplete the energy source before the ending point and finish the race with zero energy in the third phase ($E(t) = 0$, for $t \in [t_2, T]$). With these assumptions, he carried out the velocity profile and found that the velocity in the middle phase (with no pre-assumption on it) is constant. He argued that these assumptions arise from the fact that in the optimal solution, variables with inequality constraints should appear at their extreme bounds (and become equality).

II-B Reinforcement learning formulation

Our proposed algorithm for finding optimal control inputs of the runner is based on *policy gradient reinforcement learning* method thereby we use (1) a fully connected feed-forward neural network as the parameterized probabilistic controller model, and (2) a numerical simulation of the dynamical model, that is described in Subsection II-A, to simulate the runner's environment and provide the reward signal. The neural network controller model is used as the parametric policy function to transform current state of the runner into predicted optimal propulsive force for the next time step. This predicted action value is then used in the numerical simulation to apply the propulsive force, update the runner's state accordingly and compute the reward value of the new state. Then the reward value will be used eventually to update the neural network parameters using gradient-based stochastic optimization, see Figure 1 for conceptual illustration.

The RL algorithm, in general, relies on trial and error, while enforcing actions in the *good trials* by rewarding them. In our consideration of the runner's problem, for example, positions close to the finish line are highly rewarded, while any constraint violation will terminate the run with considerably smaller reward value. Utilized policy gradient method is a family of RL algorithms that parameterize the policy function directly and optimize its parameters by gradient ascent on the performance objective (cumulative reward, also known as return). This optimization is usually performed on-policy, meaning that each update only uses data collected while acting according to the most recent version of the policy. The key idea underlying policy gradient methods in RL is to push up the probabilities of actions that lead to higher return, and push down the probabilities of actions that lead to lower return, until the model reaches to the optimal policy.

In this section, we describe the controller model that predicts the propulsive force for a given state, the simulated runner and the RL optimization procedure to identify the controller model parameters.

1) Neural network control model

The neural network controller model takes runner's current state (namely instantaneous position, velocity and available energy) as input and through several parametric hidden layers generates the corresponding predicted mean value of the normal distribution on runner's propulsive

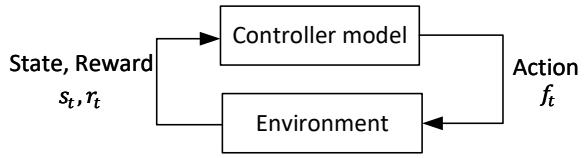


Fig. 1: Modules of reinforcement learning framework: (1) *Controller model*: reads the state and reward from current time of the environment and computes the next action f_t. Also, reward signal r_t is used to train the model, (2) *Environment*: reads the action input to advance execution of the dynamical model one step forward.

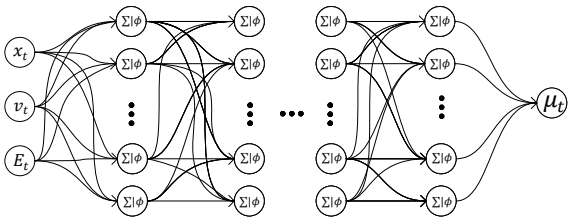


Fig. 2: Structure of a sample neural network as the predictor of the mean of the normal distribution on propulsive force, given runner's state.

force (as the next control input). Suppose that the runner's state in time t is $s_t = (x_t, v_t, E_t)$. If the corresponding output of the neural network for this input state is μ_t, then the next propulsive force is sampled from the probabilistic policy distribution of $\pi_t(f_t|s_t)$ as

$$f_t \sim \mathcal{N}(\mu_t, \sigma^2)$$

where σ^2 is assumed to be constant for all t.

Trainable parameters of the neural network are the set of weight matrices W_i and bias vectors B_i for each layer. Forward path of the neural network consists of computing the values of the stacked hidden layers by

$$z_i = W_i \cdot h_{i-1} + B_i$$

$$h_i = \tanh(z_i)$$

where $\tanh(\cdot)$ is used as the activation function. Finally, the only output of the network is weighted average of the last hidden layer values. Figure 2 shows the structure of the neural network control model.

2) Environment

In RL context, a simulated (or real) environment is needed to perform predicted action of the controller model and compute reward value of the selected action. We implemented a discretized version of dynamics of Keller's mathematical model (Equations (2) and (4)). This numerical simulation reads the instantaneous propulsive force and accordingly performs one step of update in the runner's state. We employed the Runge-Kutta method (RK4) to numerically approximate new values for the state variables (energy and velocity) based on their governing differential equations and previous values. RK4 is a forth-order iterative approximation method and uses a weighted average of four slope values (evaluation of the derivative function) at the beginning, middle and end of the time interval Δt to approximate the next value of the variable. The accumulative error in RK4 is in order of $\mathcal{O}(\Delta t^4)$ compared to first-order Euler method (Butcher, 2016) with accumulative error in order of $\mathcal{O}(\Delta t)$. Thus, RK4 will lower the approximation error caused by time discretization and reduce the sensitivity of simulation to the time step size.

The *step* method of the runner's simulation will check the success/failure conditions after each update (to check if race is finished or failed). Particularly, successful races are recognized if $x_t = D$ (or greater than), and failures are recognized if one of the followings happens: (1) predicted propulsive force exceeds its maximum possible value ($f_t > F_{max}$), (2) runner runs out of energy ($E_t < 0$), or (3) irrationally, runner moves backward ($v_t < 0$).

In addition, in the RL framework a reward signal is needed for the model optimizer to form the objective function in the training process. Therefore, while step the runner forward, simulation model will compute a reward value for every executed time step. The lower the race time, the more the reward value should be. We use the following instantaneous reward function for each state-action (step):

$$r_t(s_t, f_t) = \begin{cases} \frac{1}{t} \cdot \frac{D}{D - x_t} & \text{if } x_t < D \text{ and not } \textit{failed}, \\ \frac{1}{t} \cdot \frac{1}{\epsilon} & \text{if } x_t = D, \\ 0 & \text{if } \textit{failed}. \end{cases} \tag{6}$$

where $s_t = (x_t, v_t, E_t)$ is runner's current state, f_t is predicted action control, *failed* is true if any of the three failure constraints mentioned above is violated and ϵ is a small value. This reward function exponentially increases the reward, as the runner's position approaches the finish line and integrates the race time with inverse relation.

Given a *trajectory* (sequence of states and actions in one run) as $\delta = (s_0, f_0, s_1, f_1, ..., s_T)$, the cumulative reward (return) is defined as

$$R(\delta) = \sum_{t=0}^{T} r_t(s_t, f_t)$$

Now, if we consider the return function up to some intermediate time t as $R(\delta, t)$, then it will be proportional to the logarithmic barrier function of

$$R(\delta, t) \sim -\frac{1}{t} \cdot \log(1 - \frac{x_t}{D})$$

which helps the optimization convergence (unlike discontinuous reward functions).

Figure 3 demonstrates functions r_t and $R(\delta, t)$ over time for a successful sample trajectory of a 400-meter track. Final peak of instantaneous reward is occurred when the simulation reaches to the finish line. Also, high values in the beginning is due to the term $1/t$ and beneficial in the sense that it leads the optimization toward forward movements at the start of the race. Steep portions in the cumulative reward graph will speed up the

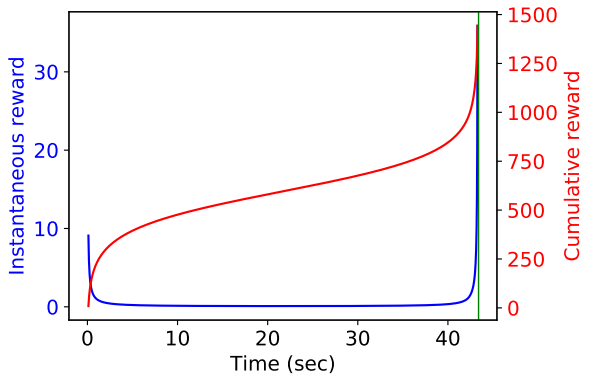


Fig. 3: Graph of (1) instantaneous reward over time (blue curve) which continuously and rapidly increases as the runner's position x_t approaches the finish line, and (2) cumulative reward over time (red curve) which is the cumulative sum of instantaneous reward values up to time t.

stochastic optimization, since each training step is based on the gradient of expected return (details in the following Subsection).

3) Optimization

The weight parameters of the controller model neural network are initialized by Kaiming uniform initialization method (He, Zhang, Ren, & Sun, 2015): $W_i \sim \mathcal{N}(0, 2/n_i)$, where n_i is the number of inputs in layer i. These model parameters need to be identified. We consider the standard policy-gradient model-free actor-critic reinforcement learning algorithm (Degris, Pilarski, & Sutton, 2012; Y. Duan, Chen, Houthooft, Schulman, & Abbeel, 2016). This algorithm maximizes the expected return function

$$J(\pi) = \mathop{\mathbb{E}}_{\delta\sim\pi}[R(\delta)] \tag{7}$$

incrementally, to identify these parameters (denote all learnable parameters by θ) of the parametric probabilistic policy $\pi_\theta(f|s)$ by applying the gradient of (7) as:

$$\theta_{k+1} = \theta_k + \alpha\nabla_\theta J(\pi_\theta)|_{\theta_k} \tag{8}$$

$$\nabla_\theta J(\pi_\theta) = \mathop{\mathbb{E}}_{\delta\sim\pi_\theta}[\sum_{t=0}^{T}\nabla_\theta \log\pi_\theta(f_t|s_t)A^{\pi_\theta}(s_t,f_t)] \tag{9}$$

where $A^{\pi_\theta}(s_t, f_t)$ is the advantage function. The advantage function measures whether or not the action is better or worse than the policy's default behavior (Schulman, Moritz, Levine, Jordan, & Abbeel, 2015) and is defined as:

$$A^{\pi_\theta}(s_t,f_t) = Q^{\pi_\theta}(s_t,f_t) - V^{\pi_\theta}(s_t) \tag{10}$$

with $Q^{\pi_\theta}(s_t, f_t)$ being the expected return when starting at state s_t, taking action f_t and then following the policy π_θ from there (on-policy action-value function) and $V^{\pi_\theta}(s_t)$ being the expected return when starting in state s_t and following the policy π_θ (Value function). These two functions are defined, formally, as:

$$Q^\pi(s,f) = \mathop{\mathbb{E}}_{\delta\sim\pi}[R(\delta)|s_0 = s, f_0 = f] \tag{11}$$

$$V^\pi(s) = \mathop{\mathbb{E}}_{\delta\sim\pi}[R(\delta)|s_0 = s] \tag{12}$$

In practice, the expected values are estimated with a sample mean. If we collect a set of trajectories $\mathcal{D} = \{\delta_i\}_{i=1,\ldots,N}$ (along with their return values) where each is obtained by employing policy π_θ in running the environment, the policy gradient is estimated with

$$\hat{g} = \frac{1}{N}\sum_{\delta\in\mathcal{D}}\sum_{t=0}^{T}\nabla_\theta \log\pi_\theta(f_t|s_t)\hat{A}(s_t,f_t) \tag{13}$$

The advantage function is also estimated by training a separate neural network to predict the value function $V^\pi(s)$. Training this model is performed along with training the policy by minimizing the mean squared residual error between estimated value for $V^\pi(s)$ and computed return value in training data.

Essentially, the training procedure consists of repetitively performing these operations: (1) collect training data $\mathcal{D}$ by using current policy's parameters θ_k and running the environment, (2) compute estimated value function $\hat{V}(s)$ for training data, (3) compute estimated advantage function $\hat{A}(s, f)$ for training data, (4) compute estimated gradient by Equation 13, (5) update the policy's parameters (and also value function estimation model's) by applying the gradient ascent (descent) step in Equation 8, and repeat the procedure from (1). For more detail about the RL training procedure see Achiam (2018); Mnih et al. (2016).

III VALIDATION EXPERIMENTS

III-A Setup

We trained separate neural network controller models for three distances 400, 1500 and 10000 meters. The fitted physiological parameters $\tau = 0.892s$, $F_{max} = 12.2m/s^2$, $\sigma = 41.54J/kg\cdot s$ and $E_0 = 2405.8J/kg$ are used from Keller (1973) to simulate a replication of the same dynamical model. Therefore, results in our model and Keller's theoretical results are comparable. A one-layer neural network with 32 neurons in its hidden layer is used (with total of 161 learnable parameters) for both control input model and value function estimators. The model is trained for 40000 steps, in each step with a batch of 5000 training data samples (collected by running the environment with current policy). Adam optimizer (Kingma & Ba, 2014) has been employed for the stochastic optimization of neural network parameters. A fixed step size is used to for the numerical simulation of the dynamical model (0.1 seconds for 400-meter, 1 second for 1500- and 10000-meter track). The value for the normal distribution variance σ^2 has been set to 0.36.

III-B Results and discussion

Figure 4 shows the progress of training procedure for 400-meter-long track over training steps. Top curve shows the trajectory return value averaged in a batch over training steps, middle curve shows trajectory lengths

averaged in a batch, and the bottom curve shows the value of State-Value function averaged in a batch. The average trajectory length (middle curve) approaches the optimal value (43.9 seconds for the 400-m case) more rapidly at the beginning of the training (0-5K training steps) and then optimizer tries to improve the time, while not violating any constraint. In the bottom curve, it can be seen that the controller model steers the simulated runner into states with higher values that achieve higher rewards. Each training step consists of a single update in the parameters of policy model based on the computed loss over a batch of 5000 data samples. Similar training procedure is performed for the 1500 and 10000 meters. After completion of training steps, all intermediate models have been tested for the best performance and the best race time has been found among them. The predicted mean value for the propulsive force has been used as the control action and no sampling is performed in the test phase.

Table I shows the results of best race times found by the RL-based method along with the theoretical results of Keller and their percent errors with the actual world record times. Average percent error between our RL records and world records is 2.38% (while the error of Keller's findings is 2.82%). The percent error between our RL records and Keller's records is 0.59%.

Figure 5 shows the propulsive force profile, dynamics of runner's position, velocity and available energy in race tracks with length 400, 1500 and 10000 meters (plotted in 5a, 5b and 5c respectively). Keller's solution is also plotted for comparison. The force and velocity profiles for 1500 and 10000 meters are analogous to theoretical optimal solution which shows the proposed RL method is able to find the sub-optimal solution of the dynamical system. However, interestingly the force profile for the 400-meter track (top plot in Figure 5a) is different in shape with the theoretical solution, even though both result in close race times. The shape difference is due to the fact that RL method is capable of finding more *complex* profiles, compared to Keller's solution that has a global view point of the solution with *a priori* assumptions which the race should be divided into exactly three phases. In other words, there are only two parameters in Keller's solution that specify the solution completely: the duration of the first and last phase (t_1 and t_2 in Keller (1973)), compared to 161 parameters in our RL-based controller model. Another potential reason for the different profile shapes is higher resolution of numerical simulation (smaller time step value) for 400 meter that enables fine-grained control for decreasing the velocity smoothly in the ending part of the race. Note for example the ending part (in the time interval from about 38 to 44 seconds) of energy dynamics (bottom plot in Figure 5a) that is laid close to, but not crossing zero energy. In this situation, there is risk of constraint violation when the controller model advances with a relatively large time step. This risk increases especially with the *probabilistic sampling* in the force distribution that can generate a sample propulsive force value far from the mean, therefore violating a system constraint, even with an acceptable mean value lower than maximum force.

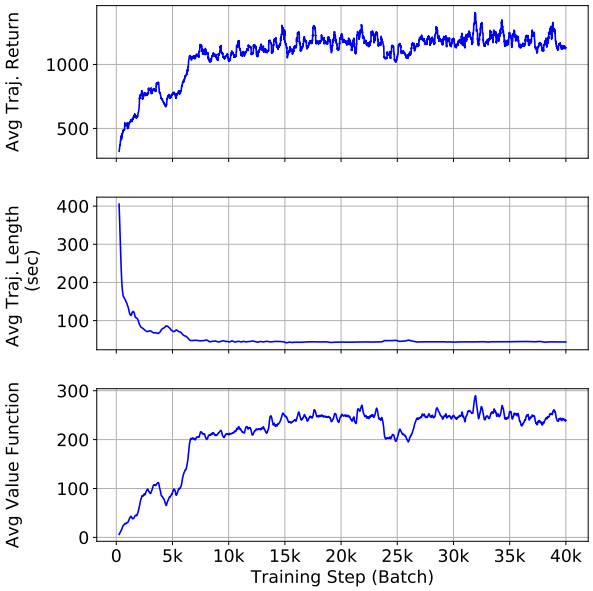


Fig. 4: Learning curves for 400-m track (return, length and evaluation of value function of states, all averaged over batch of training data).

IV CONCLUSIONS

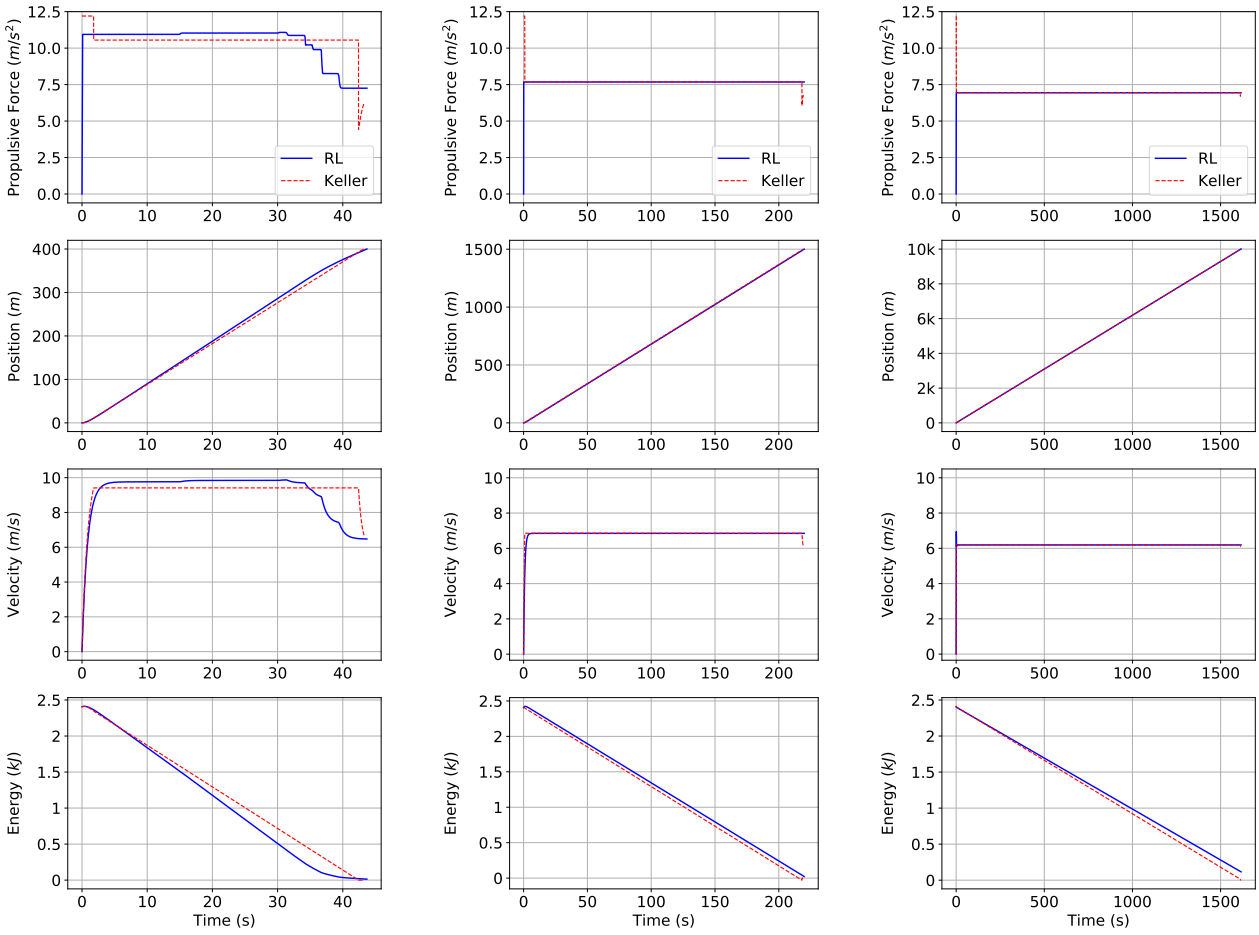
In this article, we described a reinforcement-based solution to find the optimal propulsive force profile in the running race application. The mathematical model of Keller (1973) has been employed to model the dynamics of motion and available energy in the runner's body. A numerical simulation model has been implemented for this mathematical model using Runge-Kutta approximation method, and along with a reward calculation with a designed logarithmic barrier function, has been used as the environment in the RL framework. Then, the parameters of the neural network-based probabilistic controller model have been trained by policy-gradient model-free actor-critic RL algorithm. Obtained optimal solution conforms with the theoretical results, proving the concept of the proposed method. Unlike theoretical analysis of the problem, the proposed method does not require imposing any *a priori* assumptions regarding the characteristics of the solution, enabling the method to (1) be applied on any dynamical system, and (2) find more complicated solutions in the extended search space.

Despite capability of solving the optimal control problem, the proposed RL-based method has some limitations. First, the training procedure requires to run many trials (experiences) in the simulated environment fitted to the athlete's body. This limits capability of the method in the on-line application from the scratch. However, an off-line trained model with simulated data can initialize the on-line adaptation with real-world sensory data from the runner. Another limitation of the method is that it does not guarantee the optimality of the solution which is a direct consequence of stochastic optimization used to maximize the total reward.

Future research work on the topic can focus on exploiting more realistic mathematical model for the energy

TABLE I: Race time results for 1972 world records, Keller's solution and our trained controller model (along with their percent errors). The first two data are obtained from Keller (1973).

Track Length	World Record (sec)	Keller's Theory Record (sec)	Our RL Record (sec)	Keller's Theory Error (%)	Our RL Error(%)
400 meters	44.5	43.27	43.9	-2.76	-1.34
1500 meters	213.1	219.44	219.9	2.97	3.19
10000 meters	1659.4	1614.1	1616.0	-2.72	-2.61



(a) 400-meter track (b) 1500-meter track (c) 10000-meter track

Fig. 5: Propulsive force profile (predicted with our RL method in blue and Keller's in dashed red) and resulting position, velocity and energy dynamics of the runner.

and motion dynamics (with potentially variable oxygen uptake rate and critical power during the race). Another interesting direction is to incorporate variable variance in the controller model and study the exploration behaviour in the search space due to probabilistic sampling. Here, the neural network predicts the time- and state-dependant variance σ_t^2 of the normal distribution along with its mean, regulating the exploration range during the run and potentially reduce the risk of inequality constraints violation, especially at the end of the track. Sensitivity analysis on the time step size may also be investigated in the future.

SOURCE CODE

The source code of the framework is available on: https://github.com/COMEA-TUAS/rl-optimal-control-keller

ACKNOWLEDGEMENTS

The authors gratefully acknowledge funding from Academy of Finland (ADAFI project).

REFERENCES

Abbiss, C. R., & Laursen, P. B. (2008). Describing and understanding pacing strategies during athletic competition. *Sports Medicine*, *38*(3), 239–252.

Achiam, J. (2018). Spinning up in deep reinforcement learning. *URL https://spinningup. openai. com*.

Aftalion, A. (2017). How to run 100 meters. *SIAM Journal on Applied Mathematics*, *77*(4), 1320–1334.

Aftalion, A., & Bonnans, J. F. (2014). Optimization of running strategies based on anaerobic energy and variations of velocity. *SIAM Journal on Applied Mathematics*, *74*(5), 1615–1636.

Aftalion, A., & Trélat, E. (2021). Pace and motor control optimization for a runner. *Journal of Mathematical Biology*, *83*(1), 1–21.

Alvarez-Ramirez, J. (2002). An improved peronnet-thibault mathematical model of human running performance. *European journal of applied physiology*, *86*(6), 517–525.

Behncke, H. (1997). Optimization models for the force and energy in competitive running. *Journal of mathematical biology*, *35*(4), 375–390.

Böhme, T. J., & Frank, B. (2017). Indirect methods for optimal control. In *Hybrid systems, optimal control and hybrid vehicles: Theory, methods and applications* (pp. 215–231). Cham: Springer International Publishing. doi: 10.1007/978-3-319-51317-1_7

Bonnans, F., Martinon, P., & Grélard, V. (2012). *Bocop-a collection of examples* (Unpublished doctoral dissertation). Inria.

Butcher, J. C. (2016). *Numerical methods for ordinary differential equations*. John Wiley & Sons.

Casado, A., Hanley, B., Jiménez-Reyes, P., & Renfree, A. (2021). Pacing profiles and tactical behaviors of elite runners. *Journal of Sport and Health Science*, *10*(5), 537–549.

Cooper, R. A. (1990). A force/energy optimization model for wheelchair athletics. *IEEE transactions on systems, man, and cybernetics*, *20*(2), 444–449.

Degris, T., Pilarski, P. M., & Sutton, R. S. (2012). Model-free reinforcement learning with continuous action in practice. In *2012 american control conference (acc)* (pp. 2177–2182).

Duan, J., Yi, Z., Shi, D., Lin, C., Lu, X., & Wang, Z. (2019). Reinforcement-learning-based optimal control of hybrid energy storage systems in hybrid ac–dc microgrids. *IEEE Transactions on Industrial Informatics*, *15*(9), 5355–5364.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., & Abbeel, P. (2016). Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning* (pp. 1329–1338).

Elamvazhuthi, K., & Berman, S. (2015). Optimal control of stochastic coverage strategies for robotic swarms. In *2015 ieee international conference on robotics and automation (icra)* (pp. 1822–1829).

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the ieee international conference on computer vision* (pp. 1026–1034).

Jones, A. M., Vanhatalo, A., Burnley, M., Morton, R. H., & Poole, D. C. (2010). Critical power: implications for determination of v o2max and exercise tolerance. *Medicine & Science in Sports & Exercise*, *42*(10), 1876–1890.

Kamien, M. I., & Schwartz, N. L. (2012). *Dynamic optimization: the calculus of variations and optimal control in economics and management*. courier corporation.

Keller, J. B. (1973). A theory of competitive running. *Physics today*, *26*(9), 42–47.

Keller, J. B. (1974). Optimal velocity in a race. *The American Mathematical Monthly*, *81*(5), 474–480.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Liu, B., Xie, Q., & Modiano, E. (2019). Reinforcement learning for optimal control of queueing systems. In *2019 57th annual allerton conference on communication, control, and computing (allerton)* (pp. 663–670).

Maroński, R., & Rogowski, K. (2011). Minimum-time running: a numerical approach. *Acta of Bioengineering and Biomechanics/Wrocław University of Technology*, *13*(2), 83–86.

McGibbon, K. E., Pyne, D., Shephard, M., & Thompson, K. (2018). Pacing in swimming: A systematic review. *Sports Medicine*, *48*(7), 1621–1633.

Mercier, Q., & Aftalion, A. (2020). Optimal speed in thoroughbred horse racing. *Plos one*, *15*(12), e0235024.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., ... Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *International conference on machine learning* (pp. 1928–1937).

Morton, R. H. (2006). The critical power and related whole-body bioenergetic models. *European journal of applied physiology*, *96*(4), 339–354.

Quinn, M. (2004). The effects of wind and altitude in the 400-m sprint. *Journal of sports sciences*, *22*(11-12), 1073–1081.

Rahimi, A., Dev Kumar, K., & Alighanbari, H. (2013). Particle swarm optimization applied to spacecraft reentry trajectory. *Journal of Guidance, Control, and Dynamics*, *36*(1), 307–310.

Reardon, J. (2013). Optimal pacing for running 400-and 800-m track races. *American Journal of Physics*, *81*(6), 428–435.

Schulman, J., Moritz, P., Levine, S., Jordan, M., & Abbeel, P. (2015). High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

Tan, D., & Chen, Z. (2012). On a general formula of fourth order runge-kutta method. *Journal of Mathematical Science & Mathematics Education*, *7*(2), 1–10.

Wolf, S. (2019). *Applications of optimal control to road cycling* (Unpublished doctoral dissertation).

Woodside, W. (1991). The optimal strategy for running a race (a mathematical model for world records from 50 m to 275 km). *Mathematical and computer modelling*, *15*(10), 1–12.

AUTHOR BIOGRAPHIES

SAJAD SHAHSAVARI works as Researcher at Turku University of Applied Sciences, and is a PhD candidate at University of Turku, Department of Computing, Finland.

EERO IMMONEN is an Adjunct Professor at Department of Mathematics at University of Turku, Finland, and works as Principal Lecturer at Turku University of Applied Sciences, Finland.

MASOOMEH KARAMI is a PhD candidate in Autonomous Systems Lab at University of Turku, Department of Computing, Finland.

HASHEM HAGHBAYAN is a post-doctoral researcher at University of Turku, Department of Computing, Finland.

JUHA PLOSILA is a Professor in the field of Autonomous Systems and Robotics at the University of Turku, Department of Computing, Finland.