Artif Life Robotics (2018) 23:1–9
https://doi.org/10.1007/s10015-017-0401-2

ORIGINAL ARTICLE

Robustness of linearly solvable Markov games employing inaccurate dynamics model

Ken Kinjo[1] · **Eiji Uchibe**[2] · **Kenji Doya**[1]

Received: 30 June 2017 / Accepted: 12 October 2017 / Published online: 31 October 2017

Abstract As a model-based reinforcement learning technique, linearly solvable Markov decision process (LMDP) gives an efficient way to find an optimal policy by making the Bellman equation linear under some assumptions. Since LMDP is regarded as model-based reinforcement learning, the performance of LMDP is sensitive to the accuracy of the environmental model. To overcome the problem of the sensitivity, linearly solvable Markov game (LMG) has been proposed, which is an extension of LMDP based on the game theory. This paper investigates the robustness of LMDP- and LMG-based controllers against modeling errors in both discrete and continuous state-action problems. When there is a discrepancy between the model used for building the control policy and dynamics of the tested environment, the LMG-based control policy maintained good performance while that of the LMDP-based control policy deteriorated drastically. Experimental results support the usefulness of LMG framework when acquiring an accurate model of the environment is difficult.

Keywords Model-based reinforcement learning · Linearly solvable Markov game · Linearly solvable Markov decision process · Robust control

This work was presented in part at the 19th International Symposium on Artificial Life and Robotics, Beppu, Oita, January 22–24, 2014.

✉ Eiji Uchibe
uchibe@atr.jp

[1] Okinawa Institute of Science and Technology Graduate University, Okinawa, Japan

[2] ATR Computational Neuroscience Laboratories, Kyoto, Japan

1 Introduction

In model-based reinforcement learning, an optimal controller is derived from an optimal value (cost-to-go) function by solving the Bellman equation, which is often intractable due to its nonlinearity. Linearly solvable Markov decision process (LMDP) is a computational framework to efficiently solve the Bellman equation by an exponential transformation of the value function under some constraints on action-dependent cost [11]. The LMDP framework has been applied in domains such as character control for animation [3], optimal assignment of communication resources in cellular telephone systems [8] and real-robot control [7, 12]. The major drawback of the LMDP framework is, however, that an environmental model is given in advance. Model learning is integrated with LMDP in discrete problems [2] and in continuous problems [7], but the performance of the obtained controllers is critically affected by the accuracy of the environmental model [7].

One possible way to overcome this problem is to adopt concepts from the robust control theory [9], which considers the worst adversary and derives an optimal controller using a game theoretic solution. Recently, the framework of the linearly solvable Markov game (LMG) is proposed as an extension of LMDP [4, 5], in which the optimal value function is obtained as a solution of the Hamilton–Jacobi–Isaacs (HJI) equation. Since LMG also linearizes the nonlinear HJI equation under similar assumptions of LMDP, an optimal policy can be computed efficiently. While the LMG framework has been shown to promote robustness against disturbances [4], its advantage over the LMDP framework in the face of modeling errors has not been fully investigated.

In this study, we compare the performances of the LMDP- and LMG-based controllers in the tasks of grid-world with risky states and swing-up pole. We investigate the robustness

of the controllers under variable gaps between the state transition model used for controller design and that of the actual environment. Experimental results in the discrete problem show that the LMG-based policy works well by setting the robustness parameter of LMG to the maximum value while the LMDP-based policy is very sensitive to the accuracy of the modeling error. On the contrary, experimental results in the continuous problem show that the robustness parameter should be tuned to obtain the best performance in the LMG-based policy.

2 Linearly solvable Markov game (LMG)

2.1 Markov games

Since LMDP is a special case of LMG, we provide a brief explanation of the LMG framework according to [4]. Let $x \in \mathcal{X}$ be a state of an agent, and let $u^c, u^a \in \mathcal{U}$ denote the control by the agent and disturbance by the adversary, respectively. In the Markov game, the state transition is affected by both of the agent and the adversary as $x' \sim p(x'|x, u^c, u^a)$. When x and u are continuous, $p(x'|x, u^c, u^a)$ is given by the Gaussian distribution $\mathcal{N}(x'|\boldsymbol{\mu}(x, u^c, u^a), \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ denote the mean and the covariance matrix, respectively. In particular, $\boldsymbol{\mu}$ is assumed to be:

$$\boldsymbol{\mu}(x, u^c, u^a) = \boldsymbol{a}(x) + \boldsymbol{B}(x)(u^c + u^a).$$

The agent receives immediate cost $\ell(x, u^c, u^a)$ in each step. For instance, in the first-exit case, the objective function is the expected cumulative cost [1],

$$\mathcal{J}^{\pi} = \mathbb{E}_{p(x_{t+1}|x_t, u_t^c, u_t^a)}\left[\sum_{t=0}^{\mathcal{T}_{\mathcal{A}}} \ell(x_t, u_t^c, u_t^a)\right], \tag{1}$$

where $\mathcal{T}_{\mathcal{A}}$ denotes the time when the agent arrives an absorbing state $x_{\mathcal{A}} \in \mathcal{X}_{\mathcal{A}} \subseteq \mathcal{X}$. An optimal policy, which is required to minimize the objective function while the adversary acts to maximize the objective function, is satisfied by the following HJI equation:

$$v(x) = \min_{u^c} \max_{u^a} \{\ell(x, u^c, u^a) + \mathbb{E}_{p(x'|x,u^c,u^a)}[v(x')]\}, \tag{2}$$

where $v(x)$ denotes the value function. Since Eq. (2) is a nonlinear equation due to the min and max operators, it is not trivial to find an optimal value function.

2.2 Linearization

The key trick of LMDP and LMG is to optimize the state transition probability directly instead of optimizing the policy. In other words, control and disturbance are allowed to influence the state transition probability directly. At first, a baseline state transition probability called the uncontrolled probability is introduced by

$$p^0(x'|x) = \int p(x'|x, u)\pi^0(u|x)\mathrm{d}u,$$

where $\pi^0(u|x)$ denotes a baseline policy. In LMG, a learning agent modifies a state transition probability $p^{u^c}(x'|x)$ while a disturber modifies $p^{\{u^c,u^a\}}(x'|x)$ and they are defined by:

$$p^{u^c}(x'|x) \propto g^{u^c}(x'|x)p^0(x'|x) \tag{3}$$

$$p^{\{u^c,u^a\}}(x'|x) \propto g^{u^a}(x'|x, u^c)p^{u^c}(x'|x), \tag{4}$$

where $g^{u^c}(x'|x)$ and $g^{u^a}(x'|x, u^c)$ denote the effect of control and disturbance in the state transition, respectively.

The HJI equation (2) is intractable due to its nonlinearity. However, the HJI equation is simplified by introducing the following immediate cost:

$$\begin{aligned}\ell(x, u^c, u^a) = q(x) &+ \frac{1}{\alpha}\mathcal{D}_{\alpha}\left(p^0(x'|x) \parallel p^{u^c}(x'|x)\right) \\ &- \frac{1}{\alpha}\mathrm{KL}\left(p^{\{u^c,u^a\}}(x'|x) \parallel p^{u^c}(x'|x)\right),\end{aligned} \tag{5}$$

where $q(x)$ is a state-dependent cost function. $\mathcal{D}_{\alpha}(p^0 \parallel p^{u^c})$ denotes the Rényi divergence between two probability distributions defined by:

$$\mathcal{D}_{\alpha}(p_1(x)\|p_2(x)) = \frac{\alpha}{\alpha - 1}\log\left(\int p_1(x)^{\alpha}p_2(x)^{1-\alpha}dx\right),$$

where α is called the robustness parameter ($0 \le \alpha \le 1$). The second term measures a discrepancy between the uncontrolled probability, $p^0(x'|x)$, and the controlled probability with only the control, $p^{u^c}(x'|x)$, which corresponds to control cost. The third term KL represents the Kullback–Leibler divergence between the controlled probability with only the control, $p^{u^c}(x'|x)$, and the controlled probability with the control and the disturbance, $p^{\{u^c,u^a\}}(x'|x)$. It corresponds to the cost reduction caused by the disturbance.

Under the above assumptions, the HJI equation (2) is transformed to the linear equation. If $0 \le \alpha < 1$, substituting Eq. (5) into Eq. (2) yields:

$$z(x;\alpha) = \exp((\alpha - 1)q(x))\mathbb{E}_{p^0(x'|x)}[z(x;\alpha)], \tag{6}$$

where $z(x;\alpha) = \exp((\alpha - 1)v(x))$ is called the desirability function. When $\alpha = 0$, Eq. 6 is identical to the Bellman equation linearized by LMDP [11]. If $\alpha = 1$, we obtain:

$$v(x) = q(x) + \mathbb{E}_{p^0(x'|x)}\left[v(x')\right]. \tag{7}$$

In addition, the value and desirability functions are constrained at the absorbing state, $v(x_{\mathcal{A}}) = q(x_{\mathcal{A}})$. In both cases, the control- and the disturbance-dependent cost are eliminated during the linearization. Equations (6) and (7) are linear with respect to the desirability and value functions,

respectively. According to the result of linearization, the optimal controlled probabilities are:

$$p^{*u^c}(x'|x) \propto \exp(-v(x'))p^0(x'|x), \tag{8}$$

$$p^{*\{u^a,u^c\}}(x'|x) \propto \exp((\alpha-1)v(x'))p^0(x'|x). \tag{9}$$

Thus, both the desirability function and its optimal controller of LMG and LMDP become equivalent when $\alpha = 0$.

2.3 Computing optimal policies

2.3.1 Discrete case

Equation (6) is linear with respect to the desirability function. Since it can be considered as a general eigenvalue problem, the desirability function is calculated by using a standard matrix computation package. On the other hand, Eq. (7) is linear with respect to the value function and it is regarded as a standard Bellman equation under the uncontrolled probability. The value function can be obtained by the value iteration algorithm [10].

Note that the optimal policy is not explicitly computed for the discrete setup in the LMG framework. The control policy to realize the optimal state transition probability (8) is obtained by solving the following constrained least-squares problem in each state:

$$\begin{aligned} &\min_{\pi(u|x)} \left(\sum_{u\in\mathcal{U}} \pi(u|x)p(x'|x,u) - p^{*u^c}(x'|x) \right)^2 \\ &\text{s.t.} \quad \sum_{u\in\mathcal{U}} \pi(u|x) = 1,\ 0 \le \pi(u|x) \le 1, \quad \forall\, u \in \mathcal{U}. \end{aligned} \tag{10}$$

To solve this problem, we use the function `lsqlin()` in the Matlab Optimization toolbox®.

2.3.2 Continuous case

To solve the resulting HJI equation (6) for continuous space problems, we should employ a function approximation method. According to the previous study [6, 7], the following linear function approximator is introduced:

$$z(x;\boldsymbol{w},\boldsymbol{\Theta}) = \sum_{i=1}^{N_z} w_i f(x,\boldsymbol{m}_i,\boldsymbol{S}_i), \quad \boldsymbol{\Theta} = \{(\boldsymbol{m}_i,\boldsymbol{S}_i)\}_{i=1}^{N_z}, \tag{11}$$

where $\boldsymbol{w} = (w_1, \ldots, w_{N_z})$ is the weight vector to be optimized and $f(x,\boldsymbol{m}_i,\boldsymbol{S}_i)$ is a basis function defined by:

$$f(x;\boldsymbol{m}_i,\boldsymbol{S}_i) = \exp\left(-\frac{1}{2}(x-\boldsymbol{m}_i)^{\mathrm{T}}\boldsymbol{S}_i(x-\boldsymbol{m}_i)\right),$$

where $\boldsymbol{m}_i$ and $\boldsymbol{S}_i$ denote a center position and a precision matrix of the ith basis function, respectively. w_i is a learning weight to be optimized. In the case of $\alpha = 1$, it is appropriate to approximate the value function $v(x)$ rather than the desirability function. To optimize the parameters $\boldsymbol{w}$, the least-squares method is applied for the set of collocation states $\{\boldsymbol{x}_i\}_{i=1}^{N_s}$, in which the objective function is constructed by:

$$J = \sum_{i=1}^{N_s} \left\| z(x_i;\alpha) - e^{(\alpha-1)q(x_i)}\mathbb{E}_{p^0(x'|x_i)}\left[z(x';\alpha)\right] \right\|^2.$$

See [6, 7] for more details.

Once the desirability or value function is computed, the corresponding optimal control policy $u^*(x)$ can be derived by the following equations:

$$u^*(x) = -\sigma^2\boldsymbol{B}(x)^{\mathrm{T}}\frac{1}{(\alpha-1)z(x;\alpha)}\frac{\partial z(x;\alpha)}{\partial x}, \tag{12}$$

where $\boldsymbol{B}(x)$ denotes the Jacobian matrix of the system. Then, LMG for continuous problems needs $\boldsymbol{B}(x)$ and $p^0(x'|x)$ as the environmental model explicitly. Note that the optimal action u can be computed directly in continuous problems while we need to solve the constrained least-squares problem (10) in discrete problems.

3 Discrete state-action problem

3.1 Grid-world with risky states

As an example of discrete state-action problems, we select a simple grid-world navigation problem shown in Fig. 1. When the agent steps into a risky state, it receives a high cost ($q(x) = 200$). The agent receives a small cost ($q(x) = 1$) in all other states except the goal state, where it receives zero cost and the episode is terminated. The goal of the agent is to find the shortest path to the goal state while avoiding falling off the risky states.

The state transition probability is characterized by the certainty parameter c ($0.5 \le c \le 1$), as illustrated in Fig. 1b. The agent moves in the desired direction with probability c but moves down with probability $1 - c$ due to a north wind. If the agent moves to the boundary, the agent remains in the same state. A random policy $\pi^0(u|x)$ is constructed by a discrete uniform distribution and it is used for producing the uncontrolled state transition probability $p^0(x'|x)$.

3.2 Result

Figure 2a shows the experimental results in which the optimal policy was computed in the deterministic training environment ($c = 1$). The left columns of Fig. 2a show the value functions with four different settings of the robustness parameter $\alpha \in \{0, 0.95, 0.99, 1\}$. The middle and right panels show the

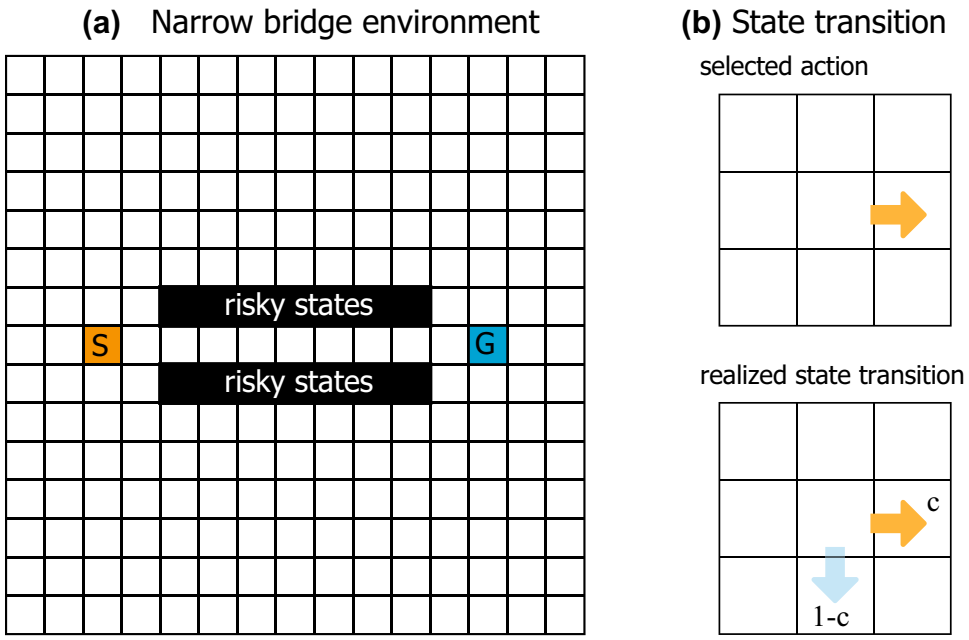

Fig. 1 Grid arrangement and state transition: **a** the start and goal states are marked with "S" and "G", respectively. The risky states exist between the start and goal states and they are colored dark-gray. **b** The agent can choose four actions: up, down, right, and left. The probability of next state depends upon certainty

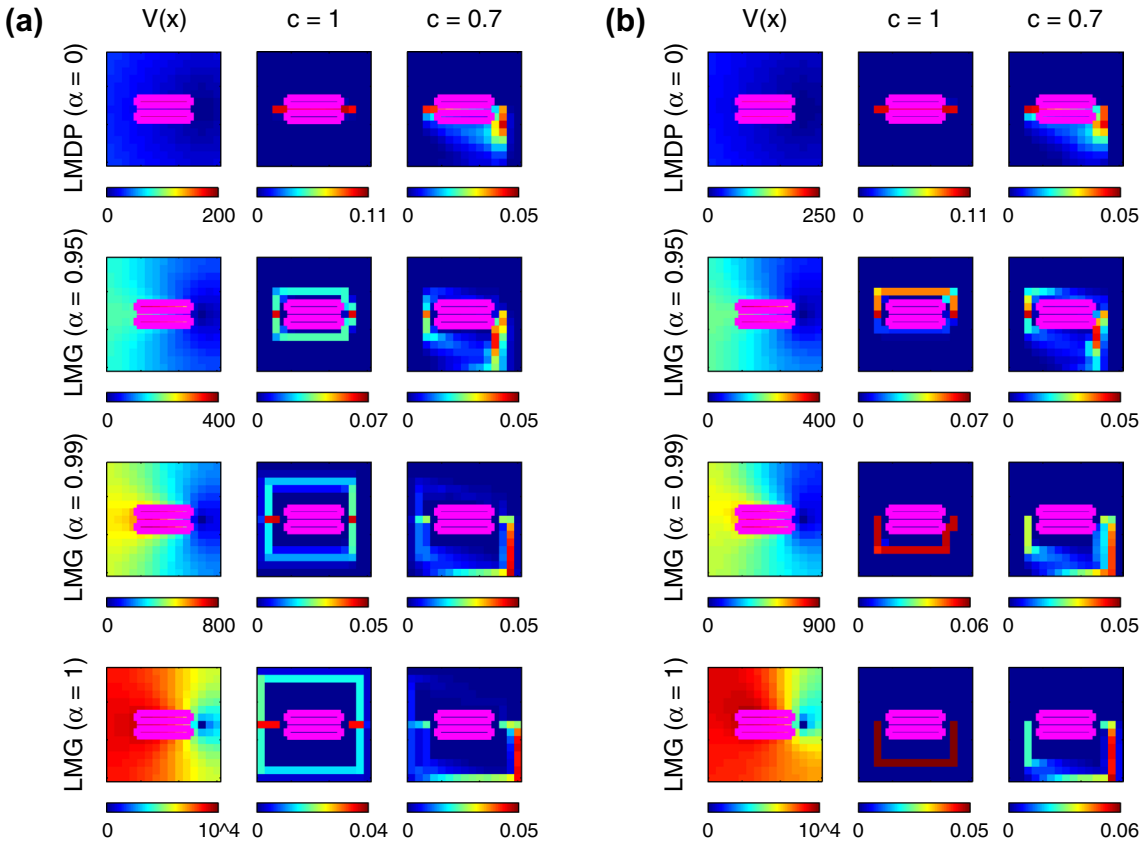

Fig. 2 Value functions and state visitation frequencies. **a** Results when the training environment is deterministic ($c = 1$). The left panels show the value functions with several setting of α. The middle and right panels show the state visitation frequencies when the test environment is deterministic ($c = 1$) and stochastic ($c = 0.7$), respectively. **b** Results when the training environment is stochastic ($c = 0.9$)

state visitation frequency which was evaluated by executing the policy in the two test environments ($c = 1$ and 0.7, respectively). The value at the entrance of the narrow path became higher as α increased, and it suggests that the agent chose the shortest path if $\alpha = 0$ while it avoided the risky states if α approached to 1, as shown in the middle panels. The right panels show that the state visitation frequency was disturbed by the north wind in the stochastic test environment. Consequently, the cumulative cost increased drastically when α was set to 0 while the most robust controller by setting $\alpha = 1$ generated similar behaviors.

Figure 2 b shows the experimental results in which the optimal policy was computed in the stochastic environment ($c = 0.9$). As compared with Fig. 2 b, the value functions were skewed due to the north wind when $\alpha \neq 0$ while the value function trained with $c = 0.9$ was the same as that with $c = 1.0$ in the case of LMDP. As a result, the LMDP-based policy preferred the shortest path even though it was computed in the stochastic environment. The reason why the stochasticity did not affect the LMDP-based policy is because $p^0(x'|x)$ is invariant with respect to c when $\pi^0(u|x)$ is uniform. In addition, the controlled probability (8) does not always satisfy the following inequality condition:

$$p^{*u^c}(x'|x) \leq \max_u p(x'|x, u).$$

In fact, we found that $p^{*u^c}(x'|x)$ for an optimal state transition (x, x') approached to 1 according to the value function. The upper panels of Fig. 2b were similar to those of Fig. 2a and it means that the stochasticity of the environment was not considered appropriately if $\alpha = 0$. On the contrary, conservative behaviors are obtained if $\alpha \geq 0.99$.

Figure 3 compares the average cumulative costs using the policies derived with five different settings of the robustness parameter α and the certainty parameter $c \in \{1, 0.9\}$ in the training environment. The performances of the policies obtained by LMDP ($\alpha = 0$) and LMG with $\alpha = 0.5$ were optimal in the deterministic environment ($c = 1$), but they deteriorated rapidly as c decreased. with the increase in the windiness, even if they were derived with an uncontrolled transition model taking into account the wind ($c = 0.9$, left panel). In contrast, the policies obtained by larger α show relatively low performance in the deterministic environment, but they performed robustly in the windier environment, even when they were derived with a windless transition model ($c = 1$, right panel).

4 Continuous state-action problem

4.1 Swing-up pole

Next, we conduct a simulation of a pole swing-up task as an example as continuous state-action problems. In the simulation, the one side of the pole was fixed and the pole could rotate in plane around the fixed point. The objective of the task is to lead the pole to an upward position and stop at this position. The continuous action u is the torque applied to the pole while the state is represented by a two-dimensional vector $x = [\theta, \omega]$, where θ and ω denote the angle and the angular velocity, respectively. The equation of motion in discrete time is modeled by:

$$\begin{aligned} \Delta x &= (\boldsymbol{a}(x) + \boldsymbol{B}(u + \sigma\xi))\Delta t \\ \boldsymbol{a}(x) &= \left[\dot{\theta},\ m\tfrac{g}{l}\sin(\theta) - \kappa\dot{\theta}\right]^{\mathrm{T}}, \quad \boldsymbol{B} = [0,\ 1]^{\mathrm{T}}, \end{aligned} \tag{13}$$

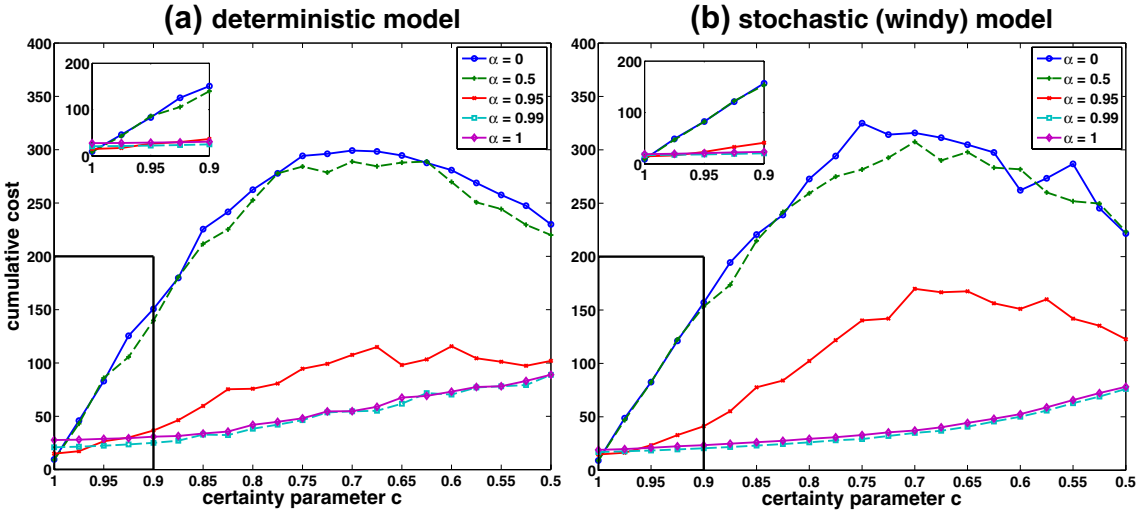

Fig. 3 Performances of different certainty. Left and right figures represent performance of the policy derived by the deterministic setting ($c = 1$) and the windy setting ($c = 0.9$), respectively. The small windows on the figures focus on the lower-left framed region. The performances are the averages over 1000 steps

where l, m, g and κ denote the length of the pole, mass, gravitational acceleration and coefficient of friction, respectively. ξ is a Gaussian noise with mean 0 and variance 1. Note that the passive dynamics $\boldsymbol{a}(x)$ is a nonlinear vector function of x while $\boldsymbol{B}$ is a constant vector. In this simulation, the physical parameters were $l = 1\,\mathrm{m}$, $g = 9.8\,\mathrm{m/s^2}$, $\kappa = 0.05$ kg $\mathrm{m^2/s}$ and $\Delta t = 10$ ms. The noise scale was set to $\sigma = 4$. The mass of the pendulum was used as a parameter to change the dynamics.

The state cost was defined so that it was zero at the goal state, using the following unnormalized Gaussian function:

$$q(x) = k(1 - \exp(x^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{cost}}^{-1}x)), \tag{14}$$

where k and $\mathrm{diag}\left(\boldsymbol{\Sigma}_{\mathrm{cost}}\right)$ denote scale of state-dependent cost and covariance matrix of Gaussian function. They are set as $k = 2.5$ and $\mathrm{diag}(\boldsymbol{\Sigma}_{\mathrm{cost}}) = [\pi/4, 4\pi/4]^2$.

The set of collocation states was uniformly distributed in the state space ($N_s = 1806$). In the simulation, only the weight parameters $\boldsymbol{w}$ are optimized. The centers $\boldsymbol{m}_i$ of the basis functions were set so as to distribute them uniformly in the state space ($N_f = 441$). On the other hand, the covariance matrix $\boldsymbol{S}_i$ was determined empirically and set to $\mathrm{diag}([\pi/20,\ \pi/20])^{-2}$.

4.2 Experimental results when the training model was different from the test model

We evaluate the robustness of the control policies obtained by the LMG framework when the test model is different from the training model. The mass of the pole was set to 1 kg in the training model while it was determined in the range (1, 2.5 kg) with the step 0.1 in the test model. For $\alpha \in \{0, 0.5, 0.75, 0.9, 0.95, 0.99, 1\}$, the desirability functions and corresponding control policies were obtained by solving the linearized HJI equation. We conducted the simulations using the obtained control policies in these test conditions. We tested 50 episodes, and each episode started from the bottom position of the pendulum and was terminated when the controller leaded to the goal position or the trial is over 100 (s) (10000 step). These results are summarized in Fig. 4.

As the mass of the pole increased, the time for swing-up increased and the success rate decreased using the control policy obtained by any value of α. In other words, the performance of all obtained control policies deteriorated. However, the deterioration rate of the control performance depended on the value of α. In the LMDP setting, $\alpha = 0$, in which the problem setting is equivalent to the LMDP framework, the performance by the obtained control policy deteriorated rapidly as the mass of the pole increased. As expected, the obtained control policy could not adapt to the change of the dynamics. On the other hand, LMG with $\alpha = 1$ could not find the robust controller as opposed to our expectations. When $\alpha = 0.75$, LMG found the most robust controller against the change of the mass of the pole in this simulation. Note that the performance was the best when $\alpha = 0.75$ even though the mass of the pole was 1 kg. As opposed to the results of the discrete problem as discussed in Sect. 3.2, the performance of the optimal controller obtained by LMDP was worse than that of the LMG-based robust controller. The reason why the LMDP-based controller failed was that the error in function approximation of the desirability and value functions was not considered. In addition, numerical errors become dominant when α is close to 1 for the continuous problems. Although the log-sum-exp technique can be used for discrete problems, it is difficult to evaluate the expectation in Eq. (1) for continuous problems. The other is because the same parameter $\boldsymbol{\Theta}$ of function approximator (11) was used although the HJI equation (7) for $\alpha = 1$ is different from Eq. (6) for $0 \leq \alpha < 1$.

4.3 Integration with model learning

Next, we investigate how the modeling error of the state transition probability affects the performance. The

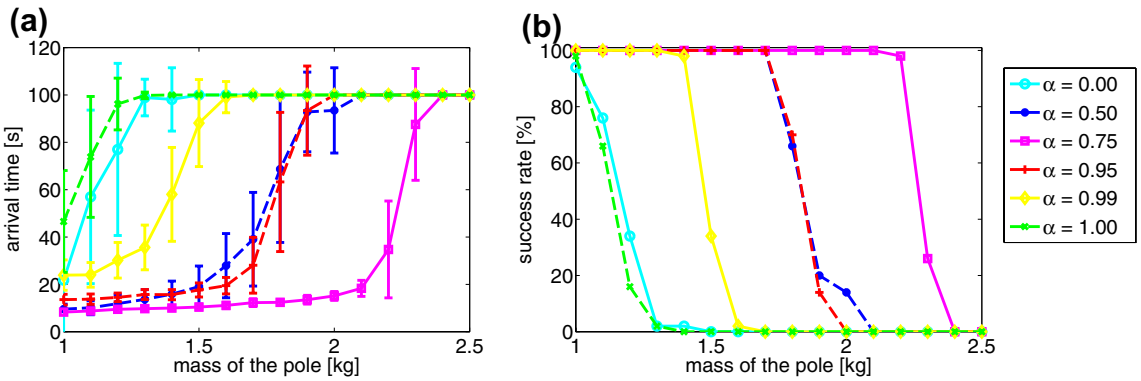

Fig. 4 **a** Arrival time to the desired state. **b** Success rate of swing-up. In each plot, each line corresponds to the mean of the trials using certain value of α and the horizontal axis is the mass of the pole in test condition

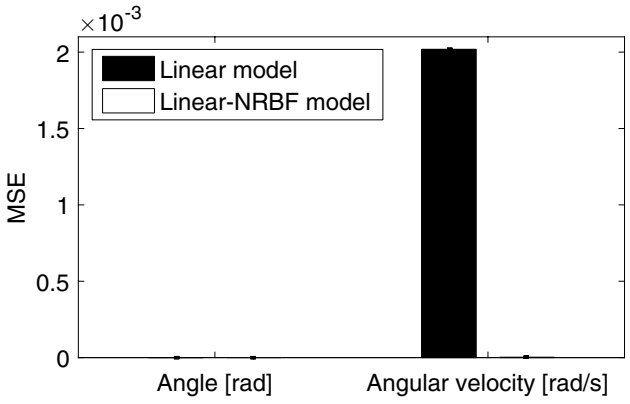

Fig. 5 The modeling error in each component: we extracted $N = 500$ samples randomly as a test data set and then calculated the approximated state transition Δx when two models were applied, respectively. After that, we computed the mean squared error (MSE) of each component

simple least-squares-based method is adopted for the method for the model learning, in which Δx in Eq. (13) are approximated by $\Delta x \approx \mathbf{W}\boldsymbol{\phi}(x, u)$, where $\boldsymbol{W}$ is a weight matrix to be optimized and $\boldsymbol{\phi}(x, u)$ is a vector consisting of basis functions. Two types of basis functions were prepared. The first was a simple linear model with respect to x and u, where $\boldsymbol{\phi}(x, u) = [x^{\mathrm{T}}, u, 1]^{\mathrm{T}}$. The other is a linear-NRBF (normalized radial basis function) model defined by $\boldsymbol{\phi}(x, u) = [x^{\mathrm{T}}, \psi_1(x), \ldots, \psi_M(x), u, 1]^{\mathrm{T}}$, where $\psi_i(x, u)$ is given by:

$$\psi_i(x) = \frac{\exp\left(-\frac{1}{2}(x-\boldsymbol{\mu}_i)^{\mathrm{T}}\boldsymbol{\Sigma}_{\psi_i}^{-1}(x-\boldsymbol{\mu}_i)\right)}{\sum_k \exp\left(-\frac{1}{2}(x-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}_{\psi_k}^{-1}(x-\boldsymbol{\mu}_k)\right)},$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_{\psi_i}$ denote the center and the covariance matrix, respectively. $\{\boldsymbol{\mu}_i\}$ were determined by K-means clustering of the training data $\mathcal{D}$ while $\{\boldsymbol{\Sigma}_{\psi_i}\}$ were tuned manually. The training data $\mathcal{D}$ were extracted from the sample data, which were acquired under the random control policy.

Figure 5 shows the MSE of the angle and angular velocity component. The angle component was approximated quite

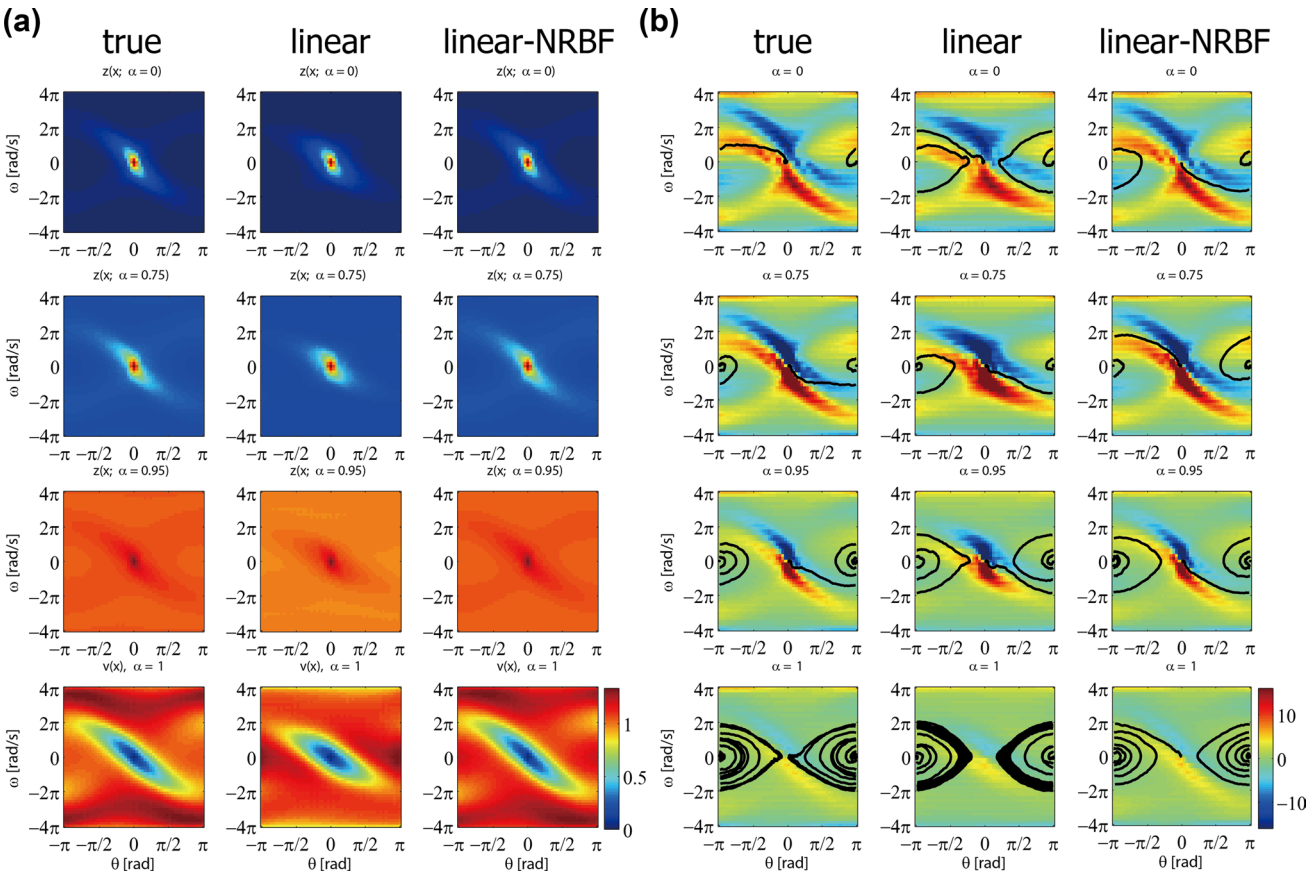

Fig. 6 **a** Solution of the linearized HJI equation. The obtained value function $v(x)$ is shown when $\alpha = 1$ and the desirability function $z(x;\alpha)$ when $0 \leq \alpha < 1$. **b** Control policy derived by Eq. (12) and plotted line on the panels are the trajectory using the resulting control policy from the bottom of pendulum. Left, middle, and right panels show the result using the true model, linear model, and linear-NRBF model, respectively

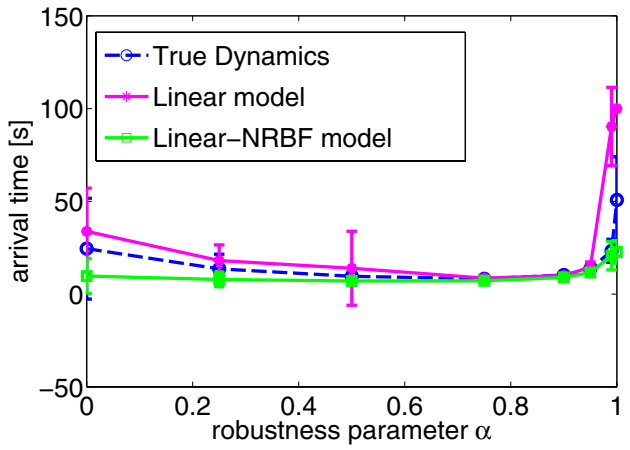

Fig. 7 The figure shows the elapsed time steps need for the swing-up. Each line corresponds to the mean of the trials using the models to solving the HJI equation and deriving the control value

accurately in both approximators because it includes only linear state transition. On the other hand, the approximation of the angular velocity component of the linear model was less accurate than that of the Liner-NRBF model because the linear model was not able to represent the nonlinear state transition.

The optimal policies derived from the models discussed above were compared with those with the true model under different levels of $\alpha \in \{0, 0.75, 0.95, 1\}$ as we did in Sect. 4.2. Figure 6a compares the estimated desirability functions of three models for $0 \leq \alpha < 1$ and the estimated value functions for $\alpha = 1$. As α became close to 1, the obtained desirability functions became flatter. This was due to the fact that the coefficient of the state cost function in Eq. (6) became small as α becomes close to 1. Since the linear-NRBF model was sufficiently accurately shown in Fig. 5, there were no significant differences in the desirability and the value function between the true and the linear-NRBF model for all α. On the other hand, the linear model produced a slightly different function. Figure 6b shows the optimal policy and typical trajectories. For the cases of true and the linear-NRBF models, the number of swinging gradually increased as α approached to 1 while the controller from the linear model showed different behaviors.

To evaluate the relationship between the control performance and the level of α in more detail, we conducted the simulations using the obtained control policies. In the simulation, we tested 50 episodes, the episodes started from the bottom position of the pendulum and were terminated when the controller leads to the goal position or the trial is over 200 (s) (20,000 steps).

The experimental results are summarized in Fig. 7. In the LMDP setting, $\alpha = 0$, the control policy obtained by the linear model which had a large approximation error took a much longer time to swing-up as compared with the other control policies. Surprisingly, the performance of the true model was slightly worse than that of the linear-NRBF model. As we discussed in Sect. 4.2, the error in function approximation should be considered even if the true model was used, the obtained control policy with $\alpha = 0$ deteriorated due to the approximation error. Nevertheless, this gap in the control performance became small as the value of α became large. Consequently, the time for swing-up was almost same among the all obtained control policies when $\alpha = 0.75, 0.9$. However, the gap became large again when α was more than 0.9. When $\alpha = 1$, the linear model showed the worse performance.

5 Conclusion

We evaluated the robustness of the control policies obtained by LMDP and LMG using the discrete and continuous state-action tasks. Our simulation results suggest that LMDP was useful only if the environment is regarded as deterministic and discrete, while LMG was efficient even if the training environment was different from the test environment. Furthermore, according to the results of the swing-up task, the performance of the LMG-based controller was improved by choosing α appropriately when the inaccurate linear model was used to approximate the environmental dynamics.

As we discussed, the error in function approximation was important in the continuous problems while the desirability function can be computed precisely for the discrete problems. In the case of the discrete problems, the desirability function can be exactly represented by a tabular representation, and it is computed by solving a generalized eigenvalue decomposition that is numerically stable. Therefore, it was appropriate to choose the largest value, $\alpha = 1$, to obtain the control policy, which is robust for the modeling error. On the other hand, in the case of the continuous problems, we should consider the effect of approximation and estimation errors of the desirability function, and therefore, the value of α should be determined carefully to obtain the robust control policy. There remains the problem of how to choose the appropriate value of α as a future work.

Acknowledgements This work was supported by JSPS/MEXT KAKENHI Grants: 17H06042 and 16K1250.

References

1. Başar T, Bernhard P (1995) H^{∞}-optimal control and related minimax design problems. A dynamic game approach. Birkhäuser, Basel
2. Burdelis MAP, Ikeda K (2014) Estimating passive dynamics distributions in linearly solvable markov decision processes from measured immediate costs in reinforcement learning problems. SICE J Control Meas Syst Integr 7(1):48–54
3. da Silva M, Durand F, Popović J (2009) Linear Bellman combination for control of character animation. ACM Trans Graph 28(3):82:1–82:10
4. Dvijotham K, Todorov E (2011) A unifying framework for linearly solvable control. In: Proceedings of the 27th annual conference on uncertainty in artificial intelligence. AUAI Press, Arlington, pp 179–186.
5. Dvijotham K, Todorov E (2012) Linearly solvable Markov games. In: Proceedings of American Control conference, pp 1845–1850
6. Todorov E (2009) Eigenfunction approximation methods for linearly-solvable optimal control problems. In Proceedings of the 2nd IEEE symposium on adaptive dynamic programming and reinforcement learning, Nashville, TN, USA, pp 161–168
7. Kinjo K, Uchibe E, Doya K (2013) Evaluation of linearly solvable Markov decision process with dynamic model learning in a mobile robot navigation task. Front Neurorobot 7(7)
8. Li A, Schrater P (2013) Efficient learning in linearly solvable MDP models. In: Proceedings of the 23rd international joint conference on artificial intelligence, pp 248–253
9. Morimoto J, Doya K (2005) Robust reinforcement learning. Neural Comput 17(2):335–359
10. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. The MIT Press, Cambridge, MA
11. Todorov E (2009) Efficient computation of optimal actions. Proc Natl Acad Sci 106(28):11478–11483
12. Uchibe E, Doya K (2014) Combining learned controllers to achieve new goals based on linearly solvable MDPs. In: Proceedings of IEEE international conference on robotics and automation