Safe Reinforcement Learning using Robust Tube-Based Model Predictive Control

Abolfazl Eskandarpour[1], Hossein Sheikhi Darani[1], and Mohammadhadi Mohandes[1]

***Abstract*— Although Model-Based Reinforcement Learning (MBRL) approaches provide desirable performance in solving complex control problems, they suffer from providing a safe exploration and exploitation in practice. In this report, we propose a safe MBRL algorithm by combining MBRL and Model Predictive Control (MPC) in which we take advantage of both to preserve safety while the closed-loop performance is optimized. The idea behind this strategy is to separate a constrained optimization problem into constraint satisfaction provided by the MPC and performance optimization using MBRL. The MPC is employed as a safety filter which enables MBRL to safely explore the dynamic uncertainty and exploit feasible actions. In fact, the action generated by the MBRL might be modified by the MPC before being applied to the real system, if it violates the constraints imposed on the system. Besides safety guarantee, the proposed method also improves the closed-loop performance compared to when we use Tube-based Linear MPC (TLMPC), Robust Nonlinear MPC (RNMPC), and MBRL to solve the problem. By simulating a pendulum and a cart-pole, the results demonstrate the effectiveness of the proposed method in safety satisfaction and performance improvement.**

I. INTRODUCTION

RL (Reinforcement learning) is known as a powerful mathematical tool for achieving experience based autonomous learning of complex and high-dimensional control policies [1]. Trial-and-error learning process is a unique feature of RL for sequential decision making tasks [2]. Recently, model-free deep RL have been shown to be capable of learning a wide range of tasks, ranging from playing video games from images [3] to learning complex locomotion skills [4].

Despite achievements [5], [6], model-free RL suffers from data inefficiency. This issue has been addressed by introducing model-based reinforcement learning. MBRL algorithms try to learn the dynamic model of the system and then extract the policy, regardless of the interactions with the environment. So this way they are generally more data efficient by extracting more information from available experiences [7]. Although alleviating data in-efficiency, facing model errors in this method is inevitable. Authors in [8] and [9] have taken this issue into consideration. In order to decrease the number of trials [10]–[12] have used GP (Gaussian Process) as a method to consider model uncertainty and learn dynamic model of the system. Additionally, A good example of a data efficient MBRL is proposed as PILCO in [10] which uses gradient-based policy search to learn control policies. Despite solving the data-inefficiency, as one of the model-free RL's limitations, PILCO has poor closed-loop feedback and misses to address the safety constraints. MBRL, itself, tries to optimize the problem over infinite horizon [13]. That's why putting constraint on MBRL algorithm is burdensome. In this regard, there are several safe-RL algorithms provided in the literature which also suffer from providing a desirable closed-loop system performance. In [14] authors propose an RL-based safe exploration algorithm which uses predefined exploration policy to produce finite set of safe states and tries to be in it's adjacency. In [15] RL agents utilizes a safety layer to stop probable unsafe actions causing unwanted short-term results, but misses to overcome the long-term ones. With the use of Markov decision processes in [16] and [17], constraint cost signal is fed to the RL agent and the goal is to keep the expected sum of the signals below a limit. The drawback with this expected sum is that we may receive random violations which we may not have considered [13].

With these limitations of RL taken into account, authors in [18] and [19] propose use of a robust MPC as a safety filter to prevent RL agent from possible unsafe actions. Since the RL agent and MPC filter are acting independently and their interaction is unilateral, RL agent may end up approaching the constraint in an oscillatory manner. In [20]authors propose use of RL in solving constrained linear quadratic regulator (LQR) for linear systems. Although these approaches may be successful, but theoretically speaking, since they deal with finite horizons, they do not guarantee safety.

Dealing with aforementioned problems, in this report we design a MBRL-MPC algorithm which guarantees a safe learning while provides a desirable performance. For the MBRL, PILCO algorithm is improved to be combined with the robust MPC approach's, i.e. TLMPC and RNMPC. In our MBRL-MPC approach, MBRL and robust MPCs meet each other at a safety filter governed by the robust MPCs. We will see that how the purposed safety filter modifies the trajectory generated by the MBRL when it violates the constraints.

The report is organized as follows: In section II the problem is formulated and in section III and IV the theory behind the proposed approach is provided. In section V the proposed MBRL-MPC algorithm is explained. Finally, in section VI the simulation results demonstrate the performance of the proposed approach.

[1] Authors are graduate students of RAMP-Lab, Faculty of Applied Science, Simon Fraser University eskandar, hsa150, mmohande@sfu.ca

II. Problem Formulation

A discrete-time deterministic dynamical system is considered as follows:

$$x(k+1) = f(x(k), u(k)) = h(x(k), u(k)) + g(x(k), u(k)), \tag{1}$$

where $h(x, u)$ and $g(x, u)$ are nominal and uncertain models of the system. Also, we assume that the system is subject to polytopic state x and control u constraints:

$$\begin{aligned} \mathcal{X} &= \{x \in \mathcal{R}^p | H_x x \leq h_x, h_x \in \mathcal{R}^{m_x}\} \\ \mathcal{U} &= \{u \in \mathcal{R}^q | H_u u \leq h_u, h_u \in \mathcal{R}^{m_u}\} \end{aligned} \tag{2}$$

where p and q are the number of states and control inputs, respectively, m_x and m_u are the number of half-spaces determine the feasible sets for state and control, respectively, i.e. $\mathcal{X}$ and $\mathcal{U}$.

Definition 1: Any predicted sate $\hat{x}(k)$ with prediction horizon N_p, i.e. $\hat{x}(k) = \{x(k+1), x(k+2), ..., x(k+N_p)\}$ is called a safe trajectory if it doesn't violate system constraints over the horizon and it belongs to a safe set, i.e. $x(k+i) \in \mathcal{X}_{safe}, i = 1, 2, ..., N_p$.

Definition 2: Assume that the system state at time instant k belongs to the safe set, i.e. $x(k) \in X_{safe}$. Any given predicted controller $\hat{u}_r(k)$ with respect to the safety policy $\pi_{safe} : \mathcal{R}^p \times \mathcal{R}^q \rightarrow \mathcal{R}^q$ is considered safe, if $x(k+i) = h(x(k+i-1), u_r(k+i-1)) \in \mathcal{X}_{safe}$ and $\hat{u}_r(k+i-1) \in \mathcal{U}_{safe}$.

Considering Definition 1 and 2, we are interested in finding a safe policy using MPC, which enables us to safely explore the dynamic of the system using MBRL while the states are maintained in the safe set. Moreover, optimal actions should be exploited using MBRL to minimize the overall cost for the task. In section III and IV we will address the required background for designing a safe MBRL algorithm using MPC.

III. Model Predictive Control

In this section we briefly introduce RNMPC and TLMPC employed in this project. A comprehensive design of the proposed approaches can be found in [21]. The proposed constrained MPC algorithms are capable of dealing with system constraints, uncertainties and external disturbances imposed on the system. Consider the predicted state $\hat{x}$ and predicted control input $\hat{u}$ over the prediction horizon N_p and control horizon N_c, a constrained quadratic optimization problem can be defined as follows:

$$\begin{aligned} \min_{\hat{u}} \quad J(k) &= \sum_{i=1}^{N_P} \hat{x}(k+i|k)^T Q\ \hat{x}(k+i|k) \\ &+ \sum_{j=0}^{N_C} \hat{u}(k+j)^T R\ \hat{u}(k+j) \\ \text{s.t.} : \hat{u} \in \mathcal{U}_{safe} \subseteq \mathcal{U}, \hat{x} \in \mathcal{X}_{safe} \subseteq \mathcal{X}, \hat{x}(k+N_p) \in \mathcal{X}_f \end{aligned} \tag{3}$$

where $\mathcal{X}_f \subseteq \mathcal{U}_{safe}$ is a terminal set which should be designed to preserve closed-loop stability and constraint satisfaction. To address this issue, we will provide two necessary definitions for designing feasible robust sets for states of the system. Before that, We assume that the system is subject to a set of external disturbance $\mathcal{W}$.

Definition 3: The one-step robust controllable set $\mathcal{S}$ is a set of states which is robustly driven to the target set $\mathcal{X}_{safe}$:

$$\begin{aligned} \mathcal{S}(\mathcal{X}_{safe}, \mathcal{W}) \triangleq \{x(k) \in \mathcal{X} : \exists\ u(k) \in \mathcal{U} \\ s.t.\ x(k+1) \in \mathcal{X}_{safe},\ \forall w \in \mathcal{W}\} \end{aligned} \tag{4}$$

Definition 4: The Robust Control Invariant (RCI) set $\mathcal{C} \subseteq \mathcal{X}$ is a set of states for which there is a controller such that the state evolution, despite the external disturbances, remains in $\mathcal{C}$ permanently:

$$\begin{aligned} x(k) \in \mathcal{C} \rightarrow \exists\ u(k) \in \mathcal{U} s.t. \\ s.t.\ x(k+1) \in \mathcal{C},\ \forall w \in \mathcal{W},\ k \rightarrow \infty\}. \end{aligned} \tag{5}$$

So, we design two Robust MPC controllers, i.e. RNMPC and TLMPC, based on the aforementioned definitions.

A. Robust Nonlinear Model Predictive Control (RNMPC)

Considering the nonlinear dynamic (1), the optimization problem (3) can be defined as follows:

$$\begin{aligned} \min_{\hat{u}} \quad J(k) \\ \text{s.t.} : \hat{u} \in \mathcal{U}_{safe} \subseteq \mathcal{U}, \hat{x} \in \mathcal{X}, \hat{x}(k+N_p) \in \mathcal{X}_f \end{aligned} \tag{6}$$

Where $\mathcal{X}_{safe}$ is one-step controllable set based on the definition 3. In addition, the terminal set $\mathcal{X}_f$ should belong to this set.

B. Tube-based Linear Model Predictive Control (TLMPC)

The discrete-time linearized model of nonlinear dynamic (1) can be represented as follows:

$$x(k+1) = Ax(k) + Bu(k) + w, \tag{7}$$

Where $w \in \mathcal{W}$ is external disturbance. So, Then the optimization problem (3) can be defined as follows:

$$\begin{aligned} \min_{\hat{u}} \quad J(k) \\ \text{s.t.} : \hat{u} \in \mathcal{U}_{safe} \subseteq \mathcal{U}, \hat{x} \in \mathcal{X}_{safe} \subseteq \mathcal{X}, \hat{x}(k+N_p) \in \mathcal{X}_f \end{aligned} \tag{8}$$

Where $\mathcal{X}_{safe}$ is RCI set based on the definition 4. Similarly, the terminal set $\mathcal{X}_f$ should be RCI and belong to $\mathcal{X}_{safe}$.

IV. Model Based Reinforcement Learning

In this section we briefly introduce the utilized MBRL approach throughout this project. We used PILCO [10], a fully Bayesian approach for efficient MBRL in continuous state and action spaces when no task-specific expert knowledge is available. PILCO, assumes dynamic systems:

$$x(k+1) = f(x(k), u_r(k)) \tag{9}$$

with continues states $x \in \mathcal{R}^p$ and controls $u_r \in \mathcal{R}^q$ and completely unknown transition dynamics f. The objective is

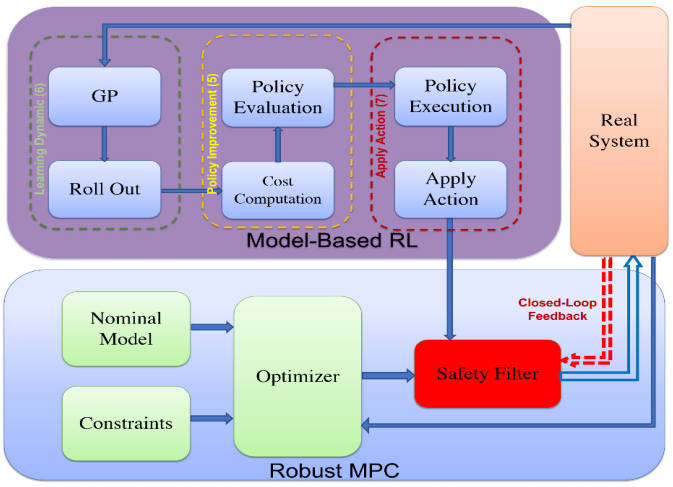

Fig. 1. Safe MBRL and Robust MPC Scheme (MBRL-MPC)

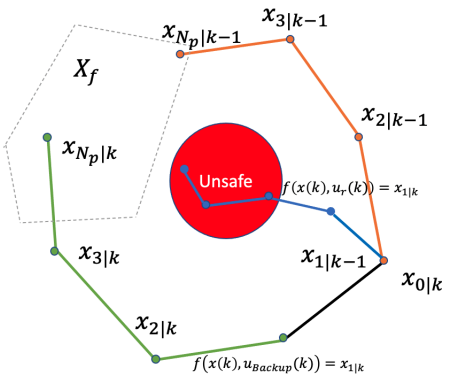

Fig. 2. Modified Action $u_r(k)$ using Backup Trajectory $u_{Backup}(k)$

to find a deterministic $policy\pi$ that minimizes the expected return:

$$J^{\pi}(\omega) = \sum_{k=0}^{K} \mathbb{E}_{x(k)}[c(x(k))] \quad , \quad x(k) \sim N(\mu_0, \Sigma_0) \quad (10)$$

of following π (parameterized by ω) for K steps, where $c(x(k))$ is the cost (negative reward) of being at state x at step k.

In the following subsections, we elaborate on key components of the PILCO: the dynamics model learning, analytic approximate policy evaluation, and gradient based policy improvement.

A. Dynamic Model Learning

PILCO utilises Gaussian Processes (GP) for its probabilistic dynamics model learning where $(x(k), u_r(k)) \in \mathcal{R}^{p+q}$ are inputs and differences $\Delta_k = x(k) - x(k-1) + \varepsilon \in \mathcal{R}^p$, $\varepsilon \sim N(0, \Sigma_\varepsilon)$, $\Sigma_\varepsilon = diag([\sigma_{\varepsilon_1}, ..., \sigma_{\varepsilon_1}])$ are targets. This GP yields to one step prediction:

$$p(x(k)|x(k-1), u(k-1)) = N(x(k)|\mu_k, \Sigma_k) \quad (11)$$

$$\mu_k = x(k-1) + \mathbb{E}_f[\Delta_k] \quad (12)$$

$$\Sigma_k = var_f[\Delta_k] \quad (13)$$

B. Policy Evaluation

For minimizing and evaluating J^{π} in Eq. (10) we have to evaluate the expected return

$$\mathbb{E}_k[c(x(k))] = \int c(x(k)) N(x(k)|\mu_k, \Sigma_k) dx(k) \quad (14)$$

for a finite horizon $k = 0, ..., K$. To this end, PILCO chooses the cost c somehow that Eq. (10) can be solved analytically:

$$c(x) = 1 - exp(-|x - x^2_{target}| \,/\, \sigma_c^2) \in [0, 1] \quad (15)$$

solving Eq. (10) also requires a long term prediction of states. To obtain the states trajectory for an finite horizon $p(x(1)), ..., p(x(k))$, PILCO cascades one-step predictions in Eqs. (11)-(13).

C. Analytic Gradients for Policy Improvement

Both μ_k and Σ_k are functionally dependent on the mean μ_u and the co-variance Σ_u of the control signal (which is applied based on ω) applied at previous state μ_{k-1} and Σ_{k-1}. Hence, PILCO can analytically computes the gradients of the expected return J^{π} with respect to the policy parameters ω by a repeated application of the chain-rule. [8] for more details.

Analytic derivatives allow for standard gradient-based non-convex optimization methods, e.g. LBFGS, which return optimized policy parameters ω^*.

The policy used in PILCO is a nonlinear RBF network:

$$\pi(x, \omega) = \sum_{i=1}^{n} w_i \phi_i(x) \quad (16)$$

$$\phi_i = exp(-0.5(x - \mu_i)^{\intercal} \Lambda^{-1} (x - \mu_i)) \quad (17)$$

with $n = 50$ squared exponential basis function centered at μ_i. PILCO considers $\omega = \{w_i, \lambda, \mu_i\} \in R^{305}$.

V. The Proposed Safe MBRL Algorithm and Contributions

In the previous sections, we briefly provided the required background to understand the theories behind the proposed method. So, in here we will more conceptually explain the design of the proposed safe MBRL algorithm using Robust MPC controllers (RNMPC and TLMPC). For the sake of brevity, we call this approach MBRL-MPC. Figure 1 gives a whole picture of the MBRL-MPC which consists of three main components, MBRL, Robust MPC and a Real System, which are fused in one entity. Based on section IV, we employed PILCO algorithm and modified it to be compatible with the MBRL-MPC approach. MBRL tries to explore the

dynamic model during several consecutive trials. Assume that the system was in a safe state at time instant $k-1$, then at time k, MBRL generate a new policy to exploit action $u_r(k)$ based on the updated explored dynamic model and policy from previous trials. At this stage, before applying the exploited action $u_r(k)$ to the system, this action should be analysed by a safety filter governed by the robust MPC.

At time instant $k-1$ we assume that the robust MPC generated a back up trajectory using a backup control input $u_{backup} = \hat{u}(k-1)$ based on the state and control input of the system. At time instant k, the safety filter check whether the generated action are safe or not. In fact, in this stage we use the nominal model to simulate the feasibility of those action. So the safety filter watch the evolution of the generated trajectory. If we design safety filter based on RNMPC controller, it examines whether the generate trajectory finally reaches the one-step robust controllable terminal set $\mathcal{X}_f$. Also, If we design it based on the TLMPC controller, it checks that the generate trajectory not only does finally reach the one-step robust controllable terminal set $\mathcal{X}_f$, but also will remain in safe tube generated by the TLMPC.

If it is safe to do so, the generated action $u_r(k)$ pass trough the filter without any modification. However, if the safety filter realizes any violation it modifies the action based on the u_{backup} as follows:

$$\begin{aligned} &\min_{u_r^{new}(k)} \quad J(k) = \|u_{Backup} - u_r^{new}(k)\| \\ &\text{s.t.} : \hat{u} \in \mathcal{U}_{safe} \subseteq \mathcal{U}, \hat{x} \in \mathcal{X}_{safe} \subseteq \mathcal{X}, \hat{x}(k+N_p) \in \mathcal{X}_f \end{aligned} \tag{18}$$

Figure 2 illustrates how backup trajectory modifies the violated trajectory generated by MBRL. The brown colour trajectory is the safe trajectory at time $k-1$ saved as the backup trajectory. Assume that the blue trajectory is generated by $u_r(k)$ which violates constraints. In this case, the safety filter modifies the trajectory by generating a new trajectory (green). So, $u_r^{new}(k)$ can be obtained by employing the u_{Backup}. The pseudo algorithm of MBRL-MPC is provided in Algorithm 1.

The performance of the MBRL-MPC can significantly varies based on the type of system structures, i.e. closed-loop and open-loop (Figure 1). In the closed-loop structure, the safety filter modifies the trajectory step by step after getting feedback from the real system. However, the safety filter in open-loop structure modifies whole trajectory without getting feedback from the system. We realized that in the presence of significant disturbances and uncertainties closed-loop structure provides a better performance.

A. Contributions

The contributions of the proposed MBRL-MPC are listed as follows:

- Designing MBRl-MPC using linear and nonlinear MPCs
- Designing robust algorithms under system uncertainties
- Successful implementation on Pendulum and Cart-Pole
- Improving PILCO algorithm for safe exploration and exploitation

Algorithm 1 MBRL-MPC

init: Sample controller parameters $\omega \sim N(0,1)$. Setup MPC based on nominal model. Apply random control signals and record data.
repeat
  Learn probabilistic (GP) dynamics model using all data.
  repeat
    Approximate inference for policy evaluation, get $J^{\pi}(\omega)$.
    Gradient-based policy improvement, $dJ^{\pi}(\omega)/d\omega$.
    Update parameters ω using LBFGS.
  until convergence; **return** ω^*.
  Set $\pi^* = \pi(\omega^*)$.
  for $k = 1, \ldots, K$ **do**
    $u_r = \pi^*(x_k)$
    if $\hat{x_{k+1}}$ (using nominal model) is in the unsafe region **then**
      $u_s = MPC$
    else
      $u_s = u_r$
    end if
    Apply u_s to system record data.
  end for
until task learned.

- Analyzing performance of the proposed method with merely using of MBRL and MPCs
- Providing two different structures, i.e. closed-loop and open-loop
- Improving the performance while providing a safety exploration and exploitation

VI. Simulation Results

We evaluated our safe MBRL-MPC approach on two classic optimal control tasks. We show that our proposed approach not only satisfies the safety constraints but also improves the performance of tube based MPC.

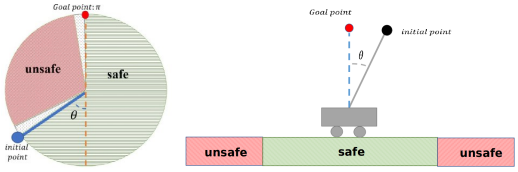

Fig. 3. a) a specific task for Pendulum Swing-up problem, b) a specific task for Cart-Pole Swing-up problem.

A. Pendulum Swing-up

The proposed method is employed to swing-up a pendulum ($x \in \mathcal{R}^2, u \in \mathcal{R}$) of length $l = 1m$ and mass $m = 1kg$, see figure 3-a. The state of the system are angle θ and angular velocity $\dot{\theta}$ of the pendulum. The pendulum angle θ is measured anti-clockwise when it is hanged down. A torque $u \in [-5, +5]$ can be applied to the pendulum. The MBRL

part doesn't have any information about pendulum's dynamic while the MPC uses the following ordinary differential equation:

$$\frac{d\mathbf{z}}{dt} = \begin{bmatrix} \frac{u - bz_1 - \frac{1}{2}mlgsinz_2}{\frac{1}{4}ml^2 + I} \\ z_1 \end{bmatrix} \tag{19}$$

where $\mathbf{z} = [\dot{\theta}, \theta]^\intercal$, $b = 0.01$ is friction coefficient.

Figure 4 depicts a whole training procedure which is completed in $N = 11$ episodes (each episode $K = 4s$). In the first two episodes a random policy is applied to generate sample inputs to learn the dynamic. Pendulum is supposed to reach $\theta = \pi$ by starting at $\theta = \pi/10$ while avoiding unsafe region $\theta \in [-\pi/10, \pi + pi/20]$. It clearly shows that MBRL-RNMPC method not only does safely prevent unsafe region during training but also reaches target by less overshoots in comparison by PILCO.

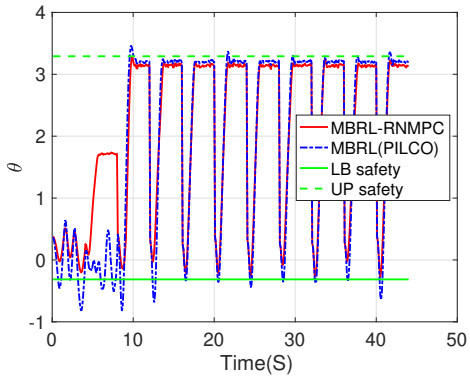

Fig. 4. Complete training episodes - pendulum.

Figure 5 illustrates the inference of task defined in Figure 3-a. As we expected PILCO learns to prefer the shortest path which violates the unsafe region, MBRL-TLMPC and MBRL-RNMPC in other hand, do not let the PILCO to guide the pendulum through unsafe region by leading it toward the longer but safe path. The ultra-fine aspect is that either MBRL-TLMPC and MBRL-RNMPC are just called two times during whole training episodes.

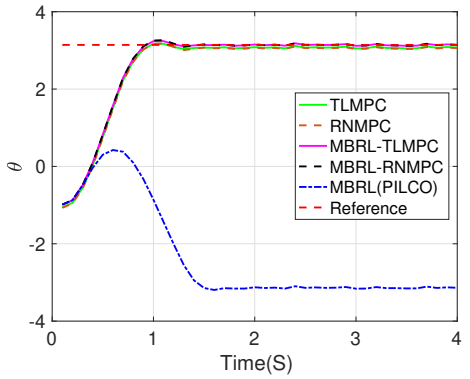

Fig. 5. Inference for task defined in Figure 3-a - pendulum.

Figure 6 is comparing the performance of different combinations of MBRL and MPC. It is subtly showing a mutually beneficial cooperation for MBRL and MPC. In one hand as mentioned earlier, MPC helps MBRL to do not violate constraints. On the other hand (which we did not expect) MBRL combination with MPC leads to better performance than pure MPC when MPC parameters are not fine-tuned very well. In this figure MBRL-RNMPC gives the best results in comparison with other shown method when considering the transition time and overshoot.

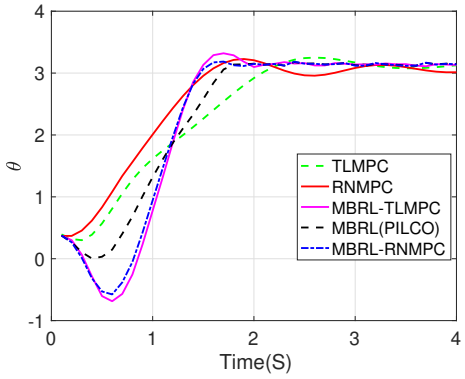

Fig. 6. Performance comparison - pendulum.

B. Cart-Pole Swing-up

We also applied our safe MBRL to control a cart-pole swing-up ($x \in \mathcal{R}^4, u \in \mathcal{R}$). The cart-pole shown in Figure 3-b. consists of a cart with mass $m_1 = 1kg$, and an attached pendulum with length $l = 0.5m$ and mass $m_2 = 0.5kg$. The state of the system is the cart's position x and velocity $\dot{x}$ in horizontal axis and also the pendulum's angle θ and angular velocity $\dot{\theta}$. A horizontal force $u \in [-10, +10]$ could be applied to the cart. Like previous experiment MBRL part has no information about cart-pole's dynamics but MPC assumes the following ordinary differential equation:

$$\frac{d\mathbf{z}}{dt} = \begin{bmatrix} z_2 \\ \frac{2m_2lz_3^2sinz_4 + 3m_2gsinz_4cosz_4 + 4u - 4bz_2}{4(m_1+m_2) - 3m_2cos^2z_4} \\ \frac{-3m_2lz_3^2sinz_4cosz_4 - 6(m_1+m_2)gsinz_4 - 6(u-bz_2)cosz_4}{4l(m_1+m_2) - 3m_2lcos^2z_4} \\ z_3 \end{bmatrix} \tag{20}$$

where $\mathbf{z} = [\mathbf{x}, \dot{\mathbf{x}}, \theta, \dot{\theta}]^\intercal$.

In the following figures we show a cart-pole swing-up task, where starting state is $\mathbf{z} = [\mathbf{0}, \mathbf{0}, \pi + \mathbf{0.2}, \mathbf{0}]^\intercal$ and the goal is to swing-up the pendulum attached to cart at $\theta = \pi$. The constraints are in the position of the cart as is shown in 3-b, so the cart position must lies in $x \in [-0.5m, 0.5m]$.

Figure 7 is showing a better performance for combination of MBRL and MPC (MBRL-RNMPC) in comparison by MBRL and MPC.

Figure 8 depicts our approach's effectiveness. PILCO violates the upper-band constraints while its combination with MPC follows the constraints. Figure 9 shows the complete

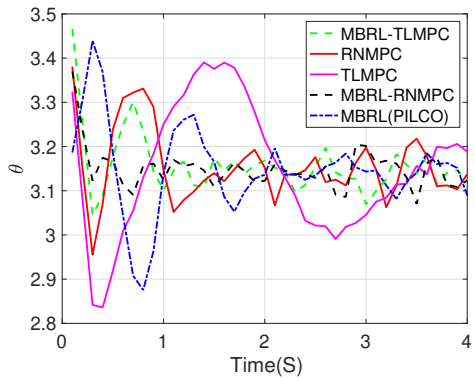

Fig. 7. Performance comparison - cart pole.

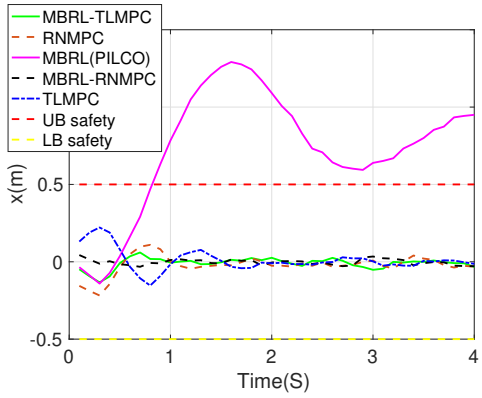

Fig. 8. Inference for task defined in Figure 3-b - cart-pole

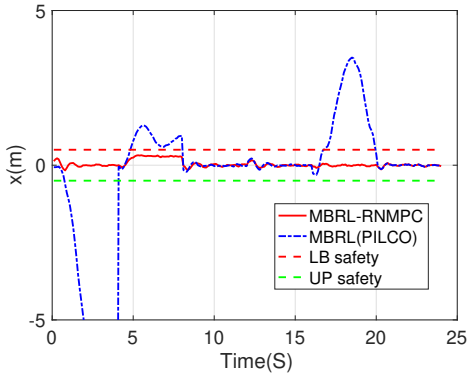

Fig. 9. Complete training episode - cart pole.

training episodes (N=6) and shows that our MBRL-RNMPC does not violates the constraints at all.

VII. Conclusion and Future works

In this report, a safe MBRL algorithm based on the robust MPC, i.e. MBRL-MPC, has been designed to deal with the problem of having a safe exploration and exploitation in practical implementations in which constraint satisfaction is of vital importance. A TLMPC and a RNMPC were designed to make the system stable under uncertainties and disturbance situations. In this regard, the required theory for designing MBRL-MPC have been provided. The main contribution of the method was to design a safety filter to modify the actions generated by MBRL when it violates the constraints imposed on the system. In the simulation section, we provided results for the implemented MBRL-MPC on the pendulum and cart-pole. We showed that MBRL-MPC is capable of safely learn the dynamic of the pendulum and cart-pole system while it avoids the restricted are during exploration. Moreover, MBRL-MPC improves the performance of the system when we just use MBRL or TLMPC and RNMPC.

Based on our experience during this project, the following issues has potential to be investigated more:

- Using stochastic MPC instead of robust MPC and Linear MPC
- Investigating the consecutive effect of the predicted trajectory using safety filter by
- changing the applied predicted trajectory length
- Utilizing a dynamic matrix weight for the MPC to improve the performance
- Employing a more complex dynamics like quadrotor
- Nominal Model improvement for MPC while the MBRL explore the real dynamic model
- Improving the dynamic learning using bootstrapping and incremental neural networks

REFERENCES

[1] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[2] S. Kamthe and M. Deisenroth, "Data-efficient reinforcement learning with probabilistic model predictive control," in *International Conference on Artificial Intelligence and Statistics*, pp. 1701–1710, PMLR, 2018.

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[4] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, pp. 1889–1897, PMLR, 2015.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[6] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[7] T. M. Moerland, J. Broekens, and C. M. Jonker, "Model-based reinforcement learning: A survey," *arXiv preprint arXiv:2006.16712*, 2020.

[8] M. Deisenroth and C. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *ICML*, 2011.

[9] J. G. Schneider, "Exploiting model uncertainty estimates for safe dynamic control learning," *Advances in neural information processing systems*, pp. 1047–1053, 1997.

[10] M. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pp. 465–472, Citeseer, 2011.

[11] M. Cutler and J. P. How, "Efficient reinforcement learning for robots using informative simulated priors," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2605–2612, IEEE, 2015.

[12] Y. Pan and E. Theodorou, "Probabilistic differential dynamic programming," *Advances in Neural Information Processing Systems*, vol. 27, pp. 1907–1915, 2014.

[13] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, "Learning-based model predictive control for safe exploration," in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 6059–6066, IEEE, 2018.

[14] J. Garcia and F. Fernández, "Safe exploration of state and action spaces in reinforcement learning," *Journal of Artificial Intelligence Research*, vol. 45, pp. 515–564, 2012.

[15] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, "Safe exploration in continuous action spaces," *arXiv preprint arXiv:1801.08757*, 2018.

[16] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International Conference on Machine Learning*, pp. 22–31, PMLR, 2017.

[17] Y. Chow, O. Nachum, A. Faust, E. Duenez-Guzman, and M. Ghavamzadeh, "Lyapunov-based safe policy optimization for continuous control," *arXiv preprint arXiv:1901.10031*, 2019.

[18] K. P. Wabersich and M. N. Zeilinger, "Safe exploration of nonlinear dynamical systems: A predictive safety filter for reinforcement learning," *arXiv preprint arXiv:1812.05506*, 2018.

[19] K. P. Wabersich and M. N. Zeilinger, "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems," *Automatica*, vol. 129, p. 109597, 2021.

[20] S. Chen, K. Saulnier, N. Atanasov, D. D. Lee, V. Kumar, G. J. Pappas, and M. Morari, "Approximating explicit model predictive control using constrained neural networks," in *2018 Annual American control conference (ACC)*, pp. 1520–1527, IEEE, 2018.

[21] B. Kouvaritakis and M. Cannon, *Model Predictive Control: Classical, Robust and Stochastic*. Advanced Textbooks in Control and Signal Processing, Springer International Publishing, 2016.