Sampling MPC

Piotr Kicki, Jan Wegrzynowski
ZNiMPRA

9 grudnia 2024

W ramach zajęć zapoznamy się z podstawowymi próbkującymi metodami sterowania. Istotnym elementem tych metod jest model który pozwala na ocenę sekwencji akcji przed ich podaniem na elementy wykonawcze robota. Funkcją określającą skuteczność naszego algorytmu jest funkcja nagrody "reward" który jest zdefiniowana wewnątrz zaproponowanych środowisk. Celem algorytmu sterującego jest jego maksymalizacja.

1 Wahadło

1. Zapoznaj się z środowiskiem Pendulum będzie ono podstawą do przetesowania kolejnych metod sterowania.

2. Spróbuj ustawić stan początkowy środowiska na zadaną wartość, np. [3.14, 0.0].

3. Zrealizuj sterownik PD w celu stabilizacji wahadła w konfiguracji pionowej skierowanej w górę, zakładając że stan początkowy to [0.1, 0.5] następnie przetestuj czy możliwe jest podniesienie wahadła ze stanu początkowego [3.1, 0.]

4. Prawdopodobnie było to niemożliwe, napisz sterownik który najpierw steruje energią do zadanego poziomu a potem przełącza się na PD w okolicy góry. Pomocna może się okazać praca.

5. Pomimo, że zadanie nie jest przesadnie skomplikowane to jego rozwiązanie wymagało wykorzystania dwóch regulatorów, z czego jednego dość specyficznego. Nie jest też oczywiste jak zaproponowane podejście ma się do funkcji nagrody zwracanej przez środowisko. Spróbujmy podejść do tego problemu w inny, bardziej ogólny sposób. Spróbujemy napisać najprostszy algorytm oparty o model próbkujący przestrzeń sterowań, którego zadaniem będzie maksymalizacja nagrody. Jego działanie można przedstawić w postaci programu:

    - ustaw aktualny stan jako stan początkowy, wygeneruj losową sekwencje sterowań o zadanej długości
    - wykorzystując metodę *step* środowiska oceń skumulowaną nagrodę całej trajektorii symulowanych stanów i sterowań

- powtarzaj n-razy krok 1, 2
- zwróć pierwszą akcje sekwencji o największej nagrodzie

6. Dla opracowanego algorytmy przygotuj wykres [heatmap] pokazujący skumulowaną nagrodą dla całego epizodu dla różnych wartości ilości rolloutów: n oraz ich długości.

7. To była prawdopodobnie najbardziej toporna metoda sterowania próbkującego opartego o model jaką można sobie wyobrazić. W dalszej części zajęć zmodyfikujemy ją trochę, tak aby otrzymać algorytm MPPI i uzyskać lepszą jakość sterowania.

   Poniżej przedstawiony jest algorytm pojedynczego wywołania algorytmu sterowania MPPI. W naszym zastosowaniu, aby nie tracić cennych obliczeń, będziemy przechowywać sekwencję sterowań $\mathbf{u}^*$ i podawać ją jako początkową sekwencję sterowań $\mathbf{u}_0$ do Algorytmu 1. Podnato, aby wziąć pod uwagę upływ czasu, przesuniemy sterowania w podanej sekwencji o jedną próbkę, pomijając pierwszą i dodając na końcu sekwencji zera.

Algorithm 1 Model Predictive Path Integral (MPPI) Control

Input: Aktualny stan x_0, początkowa sekwencja sterowań $\mathbf{u}_0$, liczba trajektorii N, horyzont symulacji T, funkcja kosztu C, dynamika systemu f, kowariancja szumu Σ, minimalne i maksymalne dopuszczalne sterowania $\underline{\mathbf{u}}, \bar{\mathbf{u}}$, temperatura λ
Output: Optymalna sekwencja sterowań $\mathbf{u}^*$
for iteration $k = 1, 2, \ldots$ **do**
  for each trajectory $n = 1, 2, \ldots, N$ **do**
    Wylosuj perturbacje sterowań $\delta\mathbf{u}^n \sim \mathcal{N}(0, \Sigma)$
    Wygeneruj zaburzone sekwencje sterowań: $\mathbf{u}^n = \mathbf{u}_k + \delta\mathbf{u}^n$
    Ogranicz zaburzone sterowania do dopuszczalnych zakresów $\bar{\mathbf{u}}^n = clamp(\mathbf{u}^n, \underline{\mathbf{u}}, \bar{\mathbf{u}})$
    Oblicz efektywne zaburzenia $\Delta\mathbf{u}^n = \bar{\mathbf{u}}^n - \mathbf{u}_k$
    Wysymuluj trajektorię $\mathbf{x}^n$ od stanu początkowego x_0, używając dynamiki f i sterowań $\mathbf{u}^n$
    Oblicz koszt trajektorii $S^n = \sum_{t=0}^{T-1} C(x_t^n, u_t^n)$
  end for
  Oblicz wagi $w^n = \exp(-S^n/\lambda)$
  Znormalizuj wagi: $\tilde{w}^n = \frac{w^n}{\sum_{m=1}^{N} w^m}$
  Zmodyfikuj sekwencję sterowań: $\mathbf{u}_{k+1} = \mathbf{u}_k + \sum_{n=1}^{N} \tilde{w}^n \Delta\mathbf{u}^n$
end for
Return: $\mathbf{u}^* = \mathbf{u}_k$