MPPI 샘플링 기법 향상을 통한 실시간 MPC

2024. 04. 12

이이수 박사님 연구실
석사과정 서윤수

Korea Institute of Science and Technology

목차

Introduction to MPPI

Why MPPI
What is MPPI

Introduction to MPPI – Why MPPI?

- **Research Goal : 비선형 모델의 무게중심 경로 생성**

	MPC	MPPI[1]
개념	모델 예측을 기반으로 시스템 상태의 최적제어 입력을 계산하는 제어 알고리즘	다중 경로 샘플링 및 확률적 경로 최적화로 최적제어 입력을 찾는 제어 알고리즘
방법론	(Deterministic) Optimal Control	Stochastic Optimal Control
모델 요구 사항	선형 모델 (NMPC는 비선형모델)	**모델 제약 없음** (비선형모델, 뉴럴넷 등)
계산 비용	높음	낮음 (샘플링 방법에 따라 차이)
GPU 적용 가능성	X	O

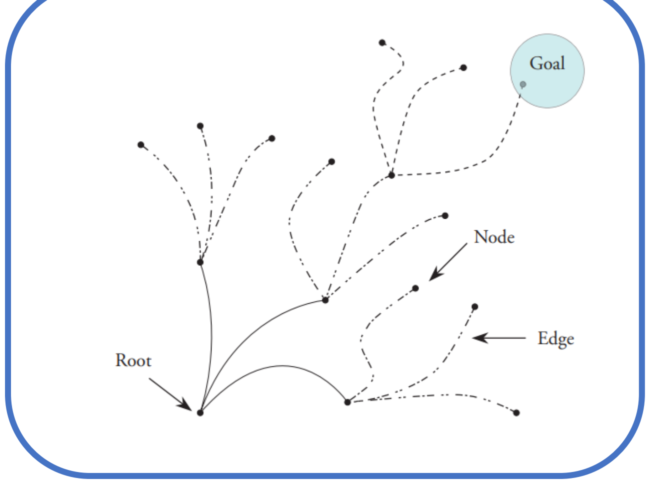
➡ **Performance 유지하며 Computation time 단축**

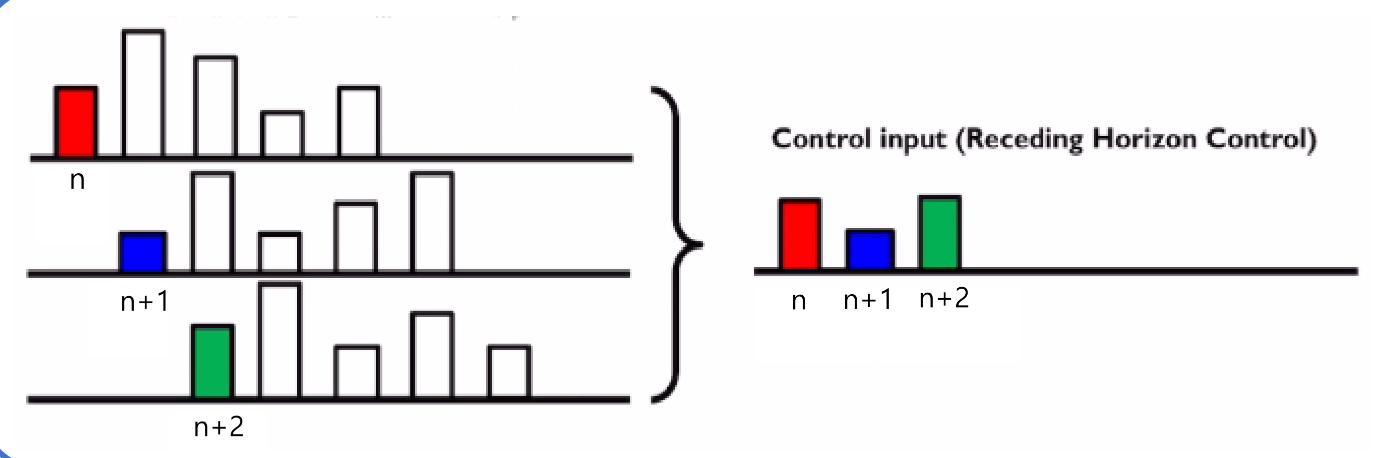
1) M. S. Gandhi "Robust Model Predictive Path Integral Control: Analysis and Performance Guarantees," in IEEE RAL

Introduction to MPPI – What is MPPI?

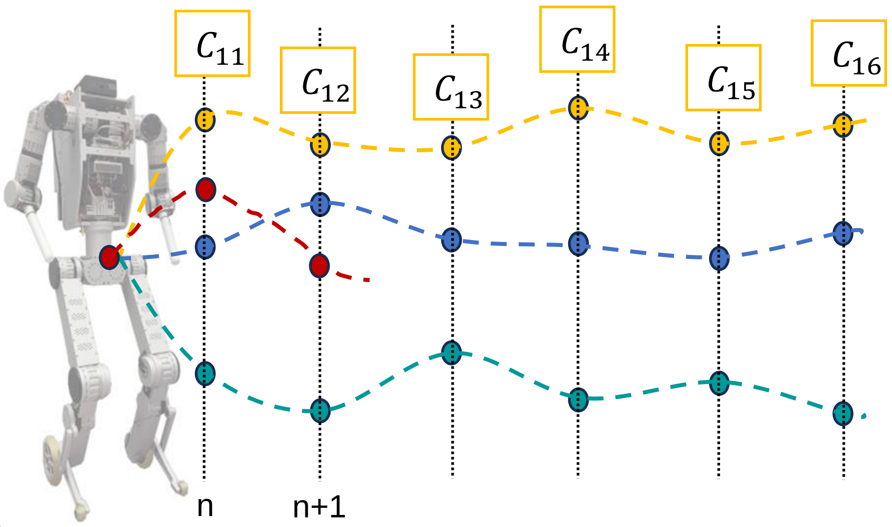
- **MPPI = SBMPC(Sampling based MPC)[1] + Path Integral**

SBMPC





Path Integral



$$C_1 = C_{11} + C_{12} + C_{13} + C_{14} + C_{15} + C_{16}$$

$$C_2 = C_{21} + C_{22} + C_{23} + C_{24} + C_{25} + C_{26}$$

$$C_3 = C_{31} + C_{32} + C_{33} + C_{34} + C_{35} + C_{36}$$

$$\frac{\sum_0^K C_{k,n} \varepsilon_{k,n}}{\sum_0^K C_{k,n}}$$

1) Caldwell "Application of sampling based model predictive control to an autonomous underwater vehicle." Ship Science and Technology

Introduction to MPPI – What is MPPI?

- **Dynamic model**

$$d\mathbf{x} = \overbrace{\mathbf{f}(\mathbf{x}_t, t)dt + \mathbf{G}(\mathbf{x}_t, t)\mathbf{u}(\mathbf{x}_t, t)dt}^{\text{Drift term}} + \underbrace{\boxed{\mathbf{B}(\mathbf{x}_t, t)d\mathbf{w}}}_{\text{Diffusion term}}$$

$$d\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t)\Delta t + \mathbf{G}(\mathbf{x}_t, t)\left(\mathbf{u}(\mathbf{x}_t, t) + \underbrace{\boxed{\frac{1}{\sqrt{\rho}}\frac{\epsilon}{\sqrt{\Delta t}}}}_{=\delta \boldsymbol{u}}\right)\Delta t \quad \text{......} \quad (1)$$

(1)의 최적 control input을 구하면[1)]

$$\mathbf{u}(\mathbf{x}_{t_i}, t_i)^* \approx \underbrace{\mathbf{u}(\mathbf{x}_{t_i}, t_i)}_{\text{기존 control input}} + \frac{\sum_{k=1}^{K} \exp\left(-\frac{1}{\lambda}\tilde{S}(\tau_{i,k})\right)\delta \boldsymbol{u}_{i,k}}{\sum_{k=1}^{K} \exp\left(-\frac{1}{\lambda}\boxed{\tilde{S}(\tau_{i,k})}\right)}$$

$$\tilde{\boldsymbol{S}}(\tau) = \phi(\boldsymbol{x}_T) + \sum_{j=1}^{N} \boxed{\tilde{q}(\boldsymbol{x}, \boldsymbol{u}, d\boldsymbol{x})}$$

$$\tilde{q}(\boldsymbol{x}, \boldsymbol{u}, d\boldsymbol{x}) = \underbrace{q(\boldsymbol{x}_t, t)}_{\text{Running cost}} + \underbrace{\frac{(1-\upsilon^{-1})}{2}\delta \boldsymbol{u}^T R \delta \boldsymbol{u} + \boldsymbol{u}^T R \delta \boldsymbol{u} + \frac{1}{2}\boldsymbol{u}^T R \boldsymbol{u}}_{\text{Importance sampling as additional cost}}$$

1) Williams, G.R., 2019. Model predictive path integral control: Theoretical foundations and applications to autonomous driving.

Introduction to MPPI – What is MPPI?

Given: K: Number of samples
N: Number of time steps
$(\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots \boldsymbol{u}_{N-1})$: Initial control sequence
Δt, $\boldsymbol{x}_{t_0}$, f, G, B, v: System/sampling dynamics
ϕ, q, R, λ: Cost parameters
$\boldsymbol{u}_{\text{init}}$: Value to initialize new controls to
while task not completed, **do**
  Generate random control variations $\delta \boldsymbol{u}$
  for $k \leftarrow 0$ **to** $K-1$, **do**
    $\boldsymbol{x} = \boldsymbol{x}_{t_0}$
    for $i \leftarrow 1$ **to** $N-1$, **do**
      $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + (f + G(\boldsymbol{u}_i + \delta \boldsymbol{u}_{i,k}))\Delta t$
      $\tilde{S}(\tau_{i+1,k}) = \tilde{S}(\tau_{i,k}) + \tilde{q}$
  for $i \leftarrow 0$ **to** $N-1$, **do**
    $\boldsymbol{u}_i \leftarrow \boldsymbol{u}_i + \left[\sum_{k=1}^{K} (\exp(-(1/\lambda)\tilde{S}_{(\tau_{i,k})})\delta \boldsymbol{u}_{i,k} / \sum_{k=1}^{K} \exp(-(1/\lambda)\tilde{S}_{(\tau_{i,k})})) \right]$
  send to actuators $(\boldsymbol{u}_0)$
  for $i \leftarrow 0$ **to** $N-2$, **do**
    $\boldsymbol{u}_i = \boldsymbol{u}_{i+1}$
  $\boldsymbol{u}_{N-1} = \boldsymbol{u}_{\text{init}}$
  Update the current state after receiving feedback
  Check for task completion

- 샘플 생성
- System 받아오기
- 샘플 control input 입력 시 cost 계산
- Cost에 따른 control input 계산
- 첫번째 최적화 input 적용

Research Objectives

Research Objectives

- **Conventional MPPI로 Performance 를 유지하며 실시간 연산 (Over 1kHz)**

Model Predictive Path Integral Control using Covariance Variable Importance Sampling 2015

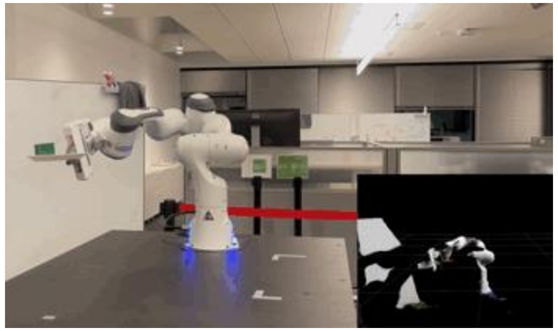
Grady Williams[1], Andrew Aldrich[1], and Evangelos A. Theodorou[1]

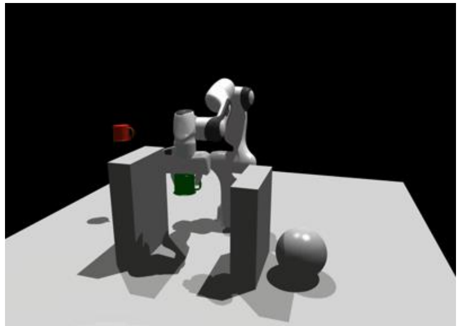
Computation time: 50Hz

STORM: An Integrated Framework for Fast Joint-Space Model-Predictive Control for Reactive Manipulation 2022

Mohak Bhardwaj[1,2], Balakumar Sundaralingam[1], Arsalan Mousavian[1], Nathan Ratliff[1], Dieter Fox[1,2], Fabio Ramos[1,3], Byron Boots[1,2]

[1]NVIDIA [2]University of Washington [3] University of Sydney





Computation time: 125Hz

Williams, G (2015). Model predictive path integral control using covariance variable importance sampling. *arXiv*

Bhardwaj, Mohak, et al. "Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation." *Conference on Robot Learning*. PMLR

Research Result

Humanoid CoM trajectory

Application – Humanoids CoM trajectory[1]

- Dynamical system : LIPM

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k$$
$$z_k = C\hat{x}_k$$

$$A = \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} \Delta t^3/6 \\ \Delta t^2/2 \\ \Delta \mathrm{t} \end{bmatrix} \quad \mathrm{C} = [1 \quad 0 \quad -\mathrm{h_c/g}]$$

- Sample : control input jerk

$$u = \dddot{x}$$

$$u_{n,k} = \sum_{0}^{N} \omega_{n,k}\varepsilon_{n,k}$$

- Cost function

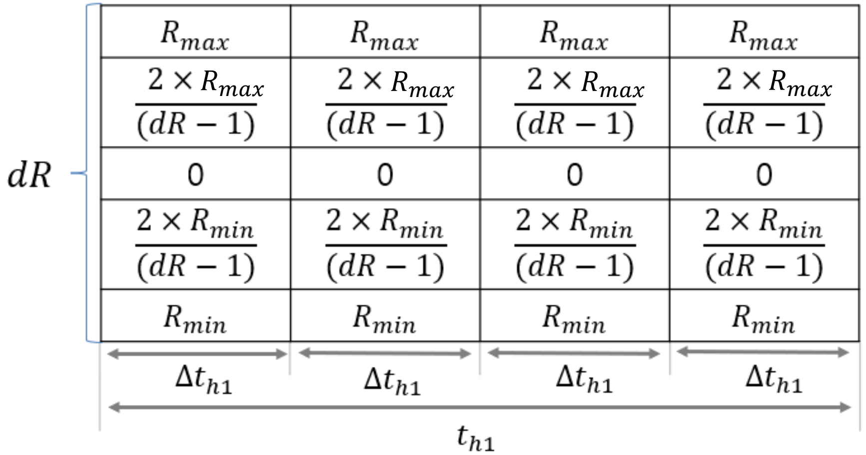
$$\hat{c} = \frac{1}{2}Q_1(Z_{k+1} - {Z_{k+1}}^{ref}) + \frac{1}{2}Q_2 u_k$$

- $\hat{x}_k$: CoM Pos,Vel,Acc
- z_k : Position of ZMP
- $u_{n,k}$: control input
- $\omega_{n,k}$: weight
- $\varepsilon_{n,k}$: sample
- N : total timestep
- k : sampling number
- n : timestep
- Q_1, Q_2: weighting parameter

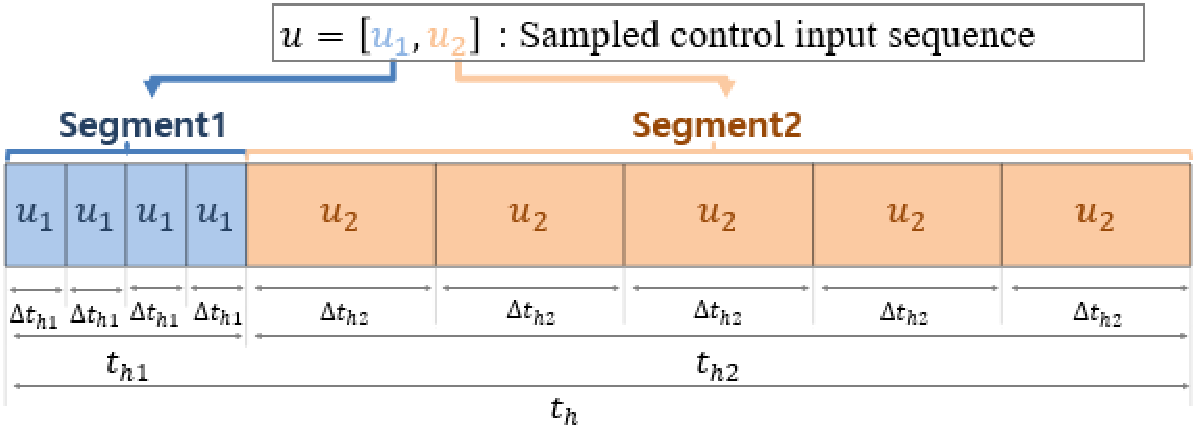
1) Y. Seo, D. Kim, J. Bak, Y. Oh and Y. Lee, "Extremely Fast Computation of CoM Trajectory Generation for Walking Leveraging MPPI Algorithm," 2023 IEEE-RAS

Application – Humanoids CoM trajectory[1]

- Single step sampling (Uniform distribution[R_{min}, R_{max}])



- Binary segmented sampling



1) Y. Seo, D. Kim, J. Bak, Y. Oh and Y. Lee, "Extremely Fast Computation of CoM Trajectory Generation for Walking Leveraging MPPI Algorithm," 2023 IEEE-RAS

Application – Humanoids CoM trajectory[1]

- Conventional MPC[2] vs MPPI

Update Frequency(Hz)	Control Method	Average Compute Time(ms)	Maximum compute Time(ms)	Average ZMP Error(m)
200	**Proposed MPPI**	**0.089**	**0.113**	**0.013**
	MPC-QP	184.02	217.32	0.001
	MPC-analytic	8.616	9.187	0.001
2000	**Proposed MPPI**	**0.064**	**0.08**	**0.011**
	Conventional MPPI	0.209	0.536	0.023

- AMC Ryzen 5 5600X 4.6GHz processor and 32 Gbyte memory
- qpOASES 사용

- Low cost board

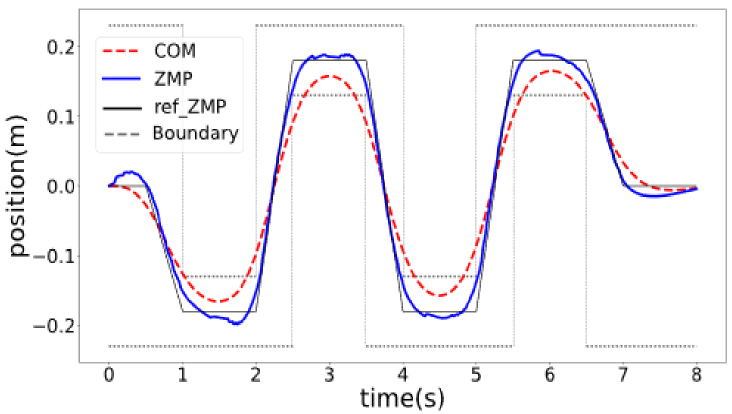
Raspberry Pi 3b
1.2 GHz processor
1GByte memory

평균 2.3ms
최대 2.5ms

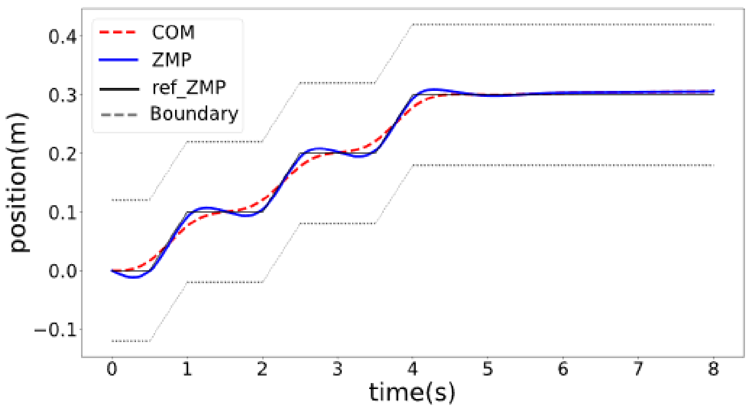
1) Y. Seo, D. Kim, J. Bak, Y. Oh and Y. Lee, "Extremely Fast Computation of CoM Trajectory Generation for Walking Leveraging MPPI Algorithm," 2023 IEEE-RAS (Humanoids),
2) P. -b. Wieber, "Trajectory Free Linear Model Predictive Control for Stable Walking in the Presence of Strong Perturbations," 2006 6th IEEE-RAS

Application – Humanoids CoM trajectory[1]

- Generated CoM trajectory



[Y coordinate result]



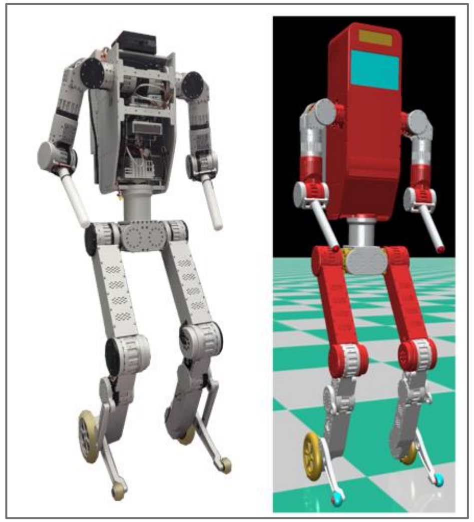
[X coordinate result]

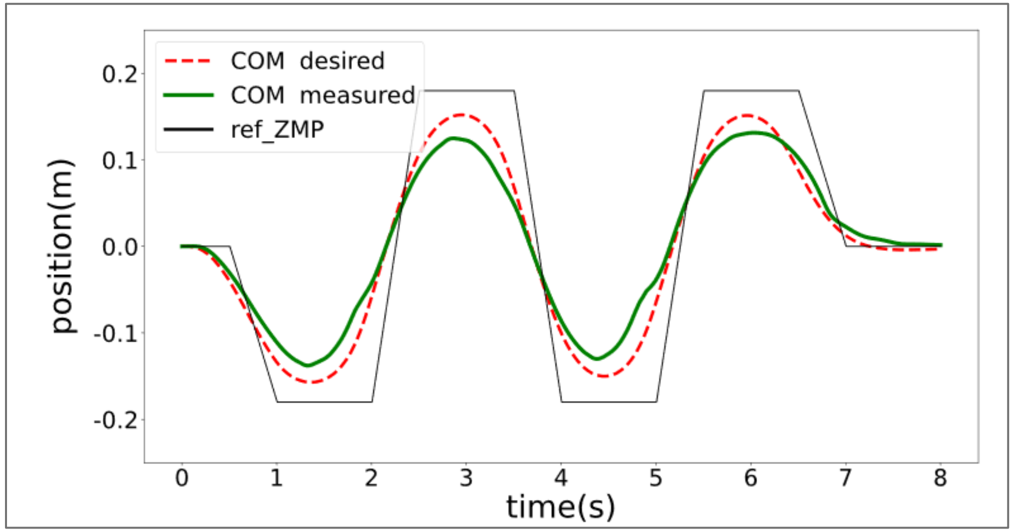
- $t_h = 0.904(s)$
- $t_{h1} = 0.004(s)$
- $t_{h2} = 0.9(s)$
- $\Delta t_{h1} = 0.0005(s)$
- $\Delta t_{h2} = 0.1(s)$
- $R_{max} = 7$
- $R_{min} = -7$
- $dR = 1$

1) Y. Seo, D. Kim, J. Bak, Y. Oh and Y. Lee, "Extremely Fast Computation of CoM Trajectory Generation for Walking Leveraging MPPI Algorithm," 2023 IEEE-RAS 22nd (Humanoids),

Application – Humanoids CoM trajectory[1]

- Simulation result





- Robot specification : 19-DoF human-sized (MAHRU-WL)
- Simulator : Mujoco(v2.0.0)

Result

평균 **0.09ms**로 풀리며 **2kHz** update 가능
라즈베리 파이에서도 **200Hz** update 가능

1) Y. Seo, D. Kim, J. Bak, Y. Oh and Y. Lee, "Extremely Fast Computation of CoM Trajectory Generation for Walking Leveraging MPPI Algorithm," 2023 IEEE-RAS 22nd (Humanoids),

Moreover Study

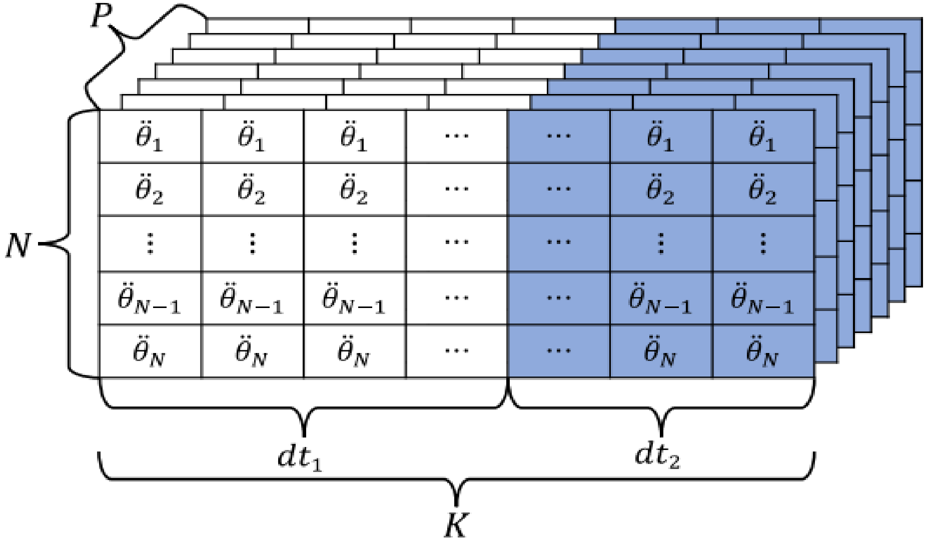
Manipulator task space control

Application – Manipulator task space control

- Dynamic time horizon

- Single step sampling (Gaussian distribution)

Sampling Table



- Constraint에 따른 cost 변화

1) Goal convergence cost

$$C_{goal} = \begin{cases} w_{pos}\tilde{e}_{pos} + w_{ori}\tilde{e}_{ori} + T^2 & \text{if } k_2 < \tilde{e}_{pos} \\ w_{pos}\tilde{e}_{pos} + w_{ori}\tilde{e}_{ori} + T & \text{if } k_1 < \tilde{e}_{pos} \leq k_2 \\ 0.5 w_{pos}\tilde{e}_{pos} + w_{ori}\tilde{e}_{ori} & \text{otherwise,} \end{cases}$$

2) Joint limit cost

$$C_{lim} = C_{lim}^{pos} + C_{lim}^{vel}$$

3) Local minima cost

$$C_{man} = w_{man}\{1 - \sqrt{\det(JJ^T)}\}$$

$$C_{cen} = w_{cen}(\theta_{cen} - \theta)^2,$$

$$\theta_{cen} = \frac{\theta_{max} - \theta_{min}}{2},$$

$$C_{local\text{-}min} = C_{man} + C_{cen}$$

4) Self collision cost (neural network)

$$C_{self\text{-}coll} = \begin{cases} k_{self\text{-}coll} & \text{if } collision \\ 0 & \text{otherwise,} \end{cases}$$

Application – Manipulator task space control

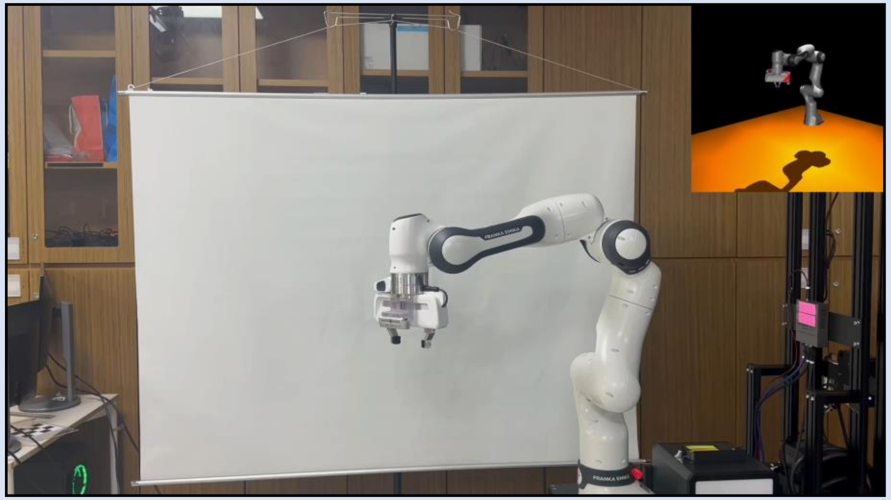
Local minima recovery

Joint limit recovery

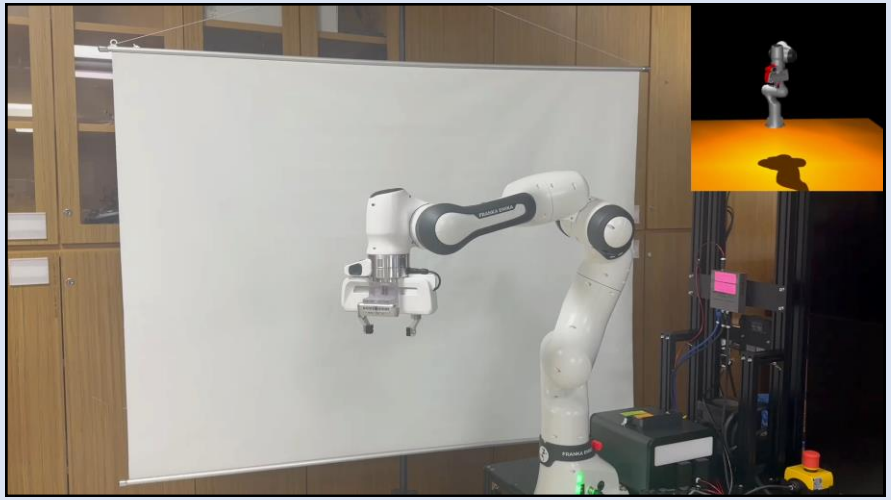
Method	Conventional MPPI(Storm)[1]	Proposed MPPI
평균 연산시간(ms)	5.53	**0.274**

1) Bhardwaj, Mohak, et al. "Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation." *Conference on Robot Learning*.

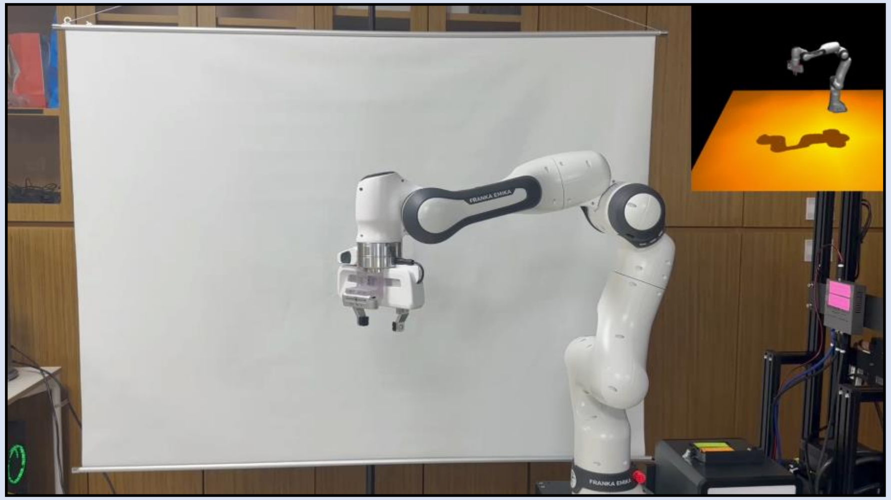
Application – Manipulator task space control



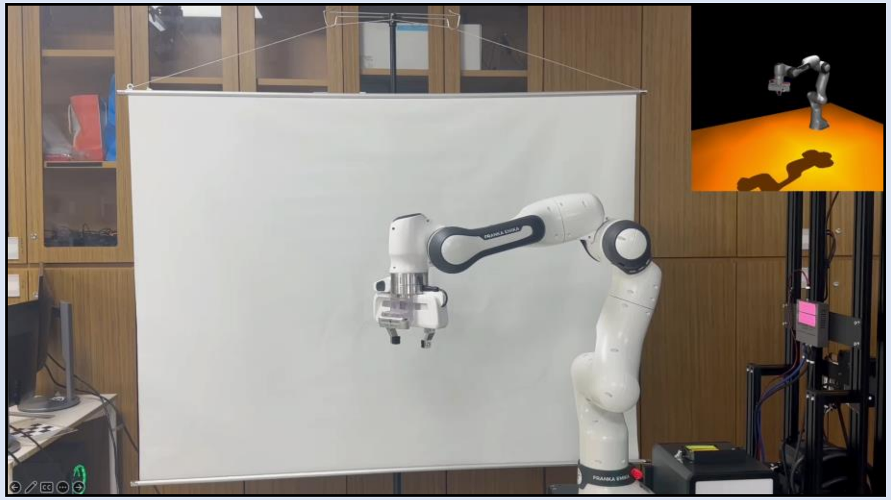
Goal Convergence cost result



Joint limit cost result



Local minima cost result



Self collision cost result

Conclusion

Summary

Future plan

Conclusion

• MPPI Summary

- Forward pass 모델 적용의 용이성
- GPU 연산 속도

• Development

- Binary segmented sampling
- Single step sampling

Sampling number ↓
연산시간 ↓

- Constraint에 따른 cost변화 → Performance ↑

• Future plan

- Real time foot planing + MPPI CoM trajectory generation
- 비선형 모델에서의 MPPI 적용

Thank you

Q&A

Single step sampling

- Motivation
  - Real time control을 위한 연산 속도를 높이기 위한 노력
  - Single-step sampling : noise sampling(control input의 변화량) 을 prediction horizon동안 동일하게 유지

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}_t, t)\mathrm{d}t + \mathbf{G}(\mathbf{x}_t, t)\mathbf{u}(\mathbf{x}_t, t)\mathrm{d}t + \mathbf{B}(\mathbf{x}_t, t)\mathrm{d}\mathbf{w}$$

$$\begin{aligned} x_{t+1} &= x_t + dx_t \\ &= x_t + f(x_t, t)\triangle t + B(x_t, t)\epsilon\sqrt{\triangle t} \\ &\qquad \epsilon \sim N(0, 1) \end{aligned}$$

$$S(\underbrace{x_0}_{\text{Initial state}}, \overbrace{x_1}^{\text{Stochastic}}, \underbrace{x_2, x_3, x_4, \ldots x_T}_{\text{Deterministic}}) = \phi(x_T) + \Sigma_{i=0}^{N} q(x_t, t)$$

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}_t, t)\mathrm{d}t + \mathbf{G}(\mathbf{x}_t, t)\overbrace{\mathbf{u}(\mathbf{x}_t, t)\mathrm{d}t}^{u(x_{t-1}) + \delta u} + \cancel{\mathbf{B}(\mathbf{x}_t, t)\mathrm{d}\mathbf{w}}$$

Single-step cost-to-go

$$S(x_0, x_1) = \phi(x_1) + \Sigma_{i=0}^{1} q(x_t, t)$$

$$\boldsymbol{x_t} = \begin{array}{ll} x_{t+0} & \\ & u_0 + \epsilon \\ x_{t+1} & \\ & u_0 + 2\epsilon \\ x_{t+2} & \\ \vdots & \\ x_{t+N} & u_0 + N\epsilon \end{array}$$

Single step sampling

- 검증
  - 전개 과정이 value function의 HJB 로부터 시작되었으므로, 수정한 single-step value function이 HJB equation을 만족하는지 확인

1. discretize 한 value function 에서 $\Delta t = T, N = 1$
2. 위 식을 continuous time space 로 변형
3. Continuous time space에서도 벨만 방정식의 원형을 유지함을 보임
   1. Stochastic HJB equation 이용가능.