Covariance Matrix Adaptation for Direct Reinforcement Learning

Freek Stulp
Cognitive Robotics Group – ENSTA-ParisTech
FLOWERS Team – INRIA/Bordeaux

Olivier Sigaud
Institut des Systèmes Intelligents et de Robotique – UPMC CNRS UMR 7222

JFPDA 23.05.2012
Motivation

- $\text{PI}^2$ powerful algorithm for direct RL on robots

Grasping under Uncertainty

After learning
Object Position -6cm

IROS’11, ICRA’11

Pick-and-Place Tasks

Humanoids’11
Motivation

- PI$^2$ powerful algorithm for direct RL on robots
  - Tuning exploration parameter in PI$^2$
Motivation

- $\text{PI}^2$ powerful algorithm for direct RL on robots
  - Tuning exploration parameter in $\text{PI}^2$

- Update rules for $\text{PI}^2$ and CEM/CMA-ES almost identical
  - But CEM and CMA-ES additionally tune exploration parameter automatically
Motivation

- $\text{PI}^2$ powerful algorithm for direct RL on robots
  - Tuning exploration parameter in $\text{PI}^2$

- Update rules for $\text{PI}^2$ and CEM/CMA-ES almost identical
  - But CEM and CMA-ES additionally tune exploration parameter automatically

Goals

- Analysis and comparison of $\text{PI}^2$/CEM/CMA-ES
- Novel algorithm $\text{PI}^2$-CMA
  - Essentially $\text{PI}^2$ with automatic exploration parameter tuning
Outline

PI

2
Outline

PI²

CEM
Outline

Reward-Weighted Averaging

- PI^2
- CEM
- CMA-ES
Outline

Reward-Weighted Averaging

- PI\(^2\)
- CEM
- CMA-ES

Comparison
Outline

Reward-Weighted Averaging

PI²
CEM
CMA-ES

Comparison
Evaluation
Task

Adaptive Exploration
Outline

**Reward-Weighted Averaging**

- PI²
- CEM
- CMA-ES

Comparison

Evaluation

Task

PI²-CMA
Outline

Reward-Weighted Averaging

- PI$^2$
- CEM
- CMA-ES

Comparison

Evaluation

Task

Adaptive Exploration

PI$^2$-CMA

Evaluation
From first principles of Stochastic Optimal Control
   Start with Hamilton Jacobi Bellman equations

1. Log transformation + benign assumption
   ⇒ Linear!

2. Transform PDE to path integral with Feynman-Kac theorem
   ⇒ Roll-outs!

3. Apply to parameterized policies
   ⇒ Model free!

⇒ Iterative update rule for $\theta$
PI\(^2\)- Algorithm

- Words
- Images
- Formulae
DMP with initial parameters $\theta$, cost function $J$.

While (cost not converged)

Explore sample exploration vectors
execute DMP
determine cost

Update
compute prob. from cost
prob.-weighted averaging
parameter update

\[ \tau_{\ddot{x}t} = \alpha \left( \beta \left( g - \dot{x}_t \right) - \dot{\omega}_t \right) + g^T_t \left( \theta + \epsilon_t, k \right) \]

\[ J(t) = \phi_{tN} + \int_{tN}^{t_f} \left( q_t + \frac{1}{2} \theta^T R \theta \right) dt \]

$\epsilon_t, k \sim N(0, \Sigma_{\epsilon})$

$\theta_k = \theta + \epsilon_k$

$\theta \leftarrow \theta + \delta \theta$

$P(\tau_{it}, k) = e^{-\frac{1}{\lambda} J(t)} \sum_{k=1}^{K} e^{-\frac{1}{\lambda} J(t)}$

$\delta \theta_{ti} = K \sum_{k=1}^{K} \left[ P(\tau_{it}, k) M_{ti, k} \epsilon_{ti, k} \right] $
**Input:** DMP with initial parameters $\theta$

$$\frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T \theta$$
**Input:** DMP with initial parameters $\theta$, cost function $J$

\[
\frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T \theta
\]

\[
J(\tau_i) = \phi_{tN} + \int_{t_i}^{t_N} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt
\]
**Input:** DMP with initial parameters $\theta$, cost function $J$

\[
\frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T \theta
\]

\[
J(\tau; i) = \phi_{tN} + \int_{t_i}^{t_N} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt
\]
- **Input**: DMP with initial parameters $\theta$, cost function $J$
- While (cost not converged)
  
  **Explore**

\[ \frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T \theta \]

\[ J(\tau_i) = \phi_{tN} + \int_{t_i}^{t_N} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt \]
- **Input**: DMP with initial parameters $\theta$, cost function $J$
- While (cost not converged)
  - **Explore**
    - sample exploration vectors
  - **Update**

\[
\frac{1}{\tau} \ddot{x}_t = \alpha(\beta(g - x_t) - \dot{x}_t) + g^T_t(\theta + \epsilon_{t,k})
\]

\[
J(\tau_i) = \phi_{tN} + \int_{t_{i}}^{t_{N}} (q_t + \frac{1}{2} \theta^T_t R\theta_t) \, dt
\]

\[
\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma^\epsilon)
\]

\[
\theta_k = \theta + \epsilon_k
\]
**Input:** DMP with initial parameters $\theta$, cost function $J$

**While (cost not converged)**

**Explore**
- sample exploration vectors
- execute DMP

**Update**

1. $\tau = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T (\theta + \epsilon_{t,k})$

2. $J(\tau_i) = \phi_{t_N} + \int_{t_i}^{t_N} (q_t + \frac{1}{2} \theta_t^T R \theta_t) dt$

$$\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma^\epsilon)$$

$$\theta_{k} = \theta + \epsilon_{k}$$
**Input:** DMP with initial parameters $\theta$, cost function $J$

While (cost not converged)

**Explore**

sample exploration vectors
execute DMP
determine cost

**Update**

$\frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T (\theta + \epsilon_{t,k})$

$J(\tau_i) = \phi_{tN} + \int_{t_i}^{tN} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt$

$\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma^\epsilon)$

$\theta_k = \theta + \epsilon_k$
- **Input**: DMP with initial parameters $\theta$, cost function $J$
- While (cost not converged)
  - **Explore**
    - sample exploration vectors
    - execute DMP
  - determine cost

\[
\begin{align*}
\frac{1}{\tau} \ddot{x}_t &= \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T (\theta + \epsilon_{t,k}) \\
J(\tau_i) &= \phi_{tN} + \int_{t_i}^{t_N} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt
\end{align*}
\]

$\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma^\epsilon)$

$\theta_k = \theta + \epsilon_k$
**Input:** DMP with initial parameters $\theta$, cost function $J$

**While** (cost not converged)

**Explore**
- sample exploration vectors
- execute DMP
- determine cost

**Update**
- compute prob. from cost

\[
\frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T (\theta + \epsilon_{t,k})
\]

\[
J(\tau) = \phi_{tN} + \int_{t_i}^{t_N} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt
\]

\[
\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma^\epsilon)
\]

\[
\theta_k = \theta + \epsilon_k
\]

\[
P(\tau_{i,k}) = \frac{e^{-\frac{1}{\lambda} J(\tau_{i,k})}}{\sum_{k=1}^{K} [e^{-\frac{1}{\lambda} J(\tau_{i,k})}]} \]
**Input**: DMP with initial parameters $\theta$, cost function $J$

While (cost not converged)

**Explore**
- sample exploration vectors
- execute DMP
- determine cost

**Update**
- compute prob. from cost
- prob.-weighted averaging

$$
\frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T (\theta + \epsilon_{t,k})
$$

$$
J(\tau_i) = \phi_{tN} + \int_{t_i}^{t_{N}} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt
$$

$$
\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma^\epsilon)
$$

$$
\theta_k = \theta + \epsilon_k
$$

$$
P (\tau_{i,k}) = \frac{e^{-\frac{1}{\lambda} \lambda J(\tau_i, k)}}{\sum_{k=1}^{K} e^{-\frac{1}{\lambda} \lambda J(\tau_i, k)}}
$$

$$
\delta \theta_{t_i} = \sum_{k=1}^{K} [P (\tau_{i,k}) M_{t_i,k} \epsilon_{t_i,k}]
$$
- **Input**: DMP with initial parameters $\theta$, cost function $J$
- While (cost not converged)

**Explore**
- sample exploration vectors
- execute DMP
- determine cost

**Update**
- compute prob. from cost
- prob.-weighted averaging
- parameter update

\[
\frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T (\theta + \epsilon_{t,k})
\]

\[
J(\tau_i) = \phi_{tN} + \int_{t_i}^{tN} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt
\]

\[
\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma^\epsilon)
\]

\[
\theta_k = \theta + \epsilon_k
\]

\[
\theta \leftarrow \theta + \delta \theta
\]

\[
P(\tau_{i,k}) = \frac{e^{-\frac{1}{\lambda} J(\tau_{i,k})}}{\sum_{k=1}^{K} \left[ e^{-\frac{1}{\lambda} J(\tau_{i,k})} \right]}
\]

\[
\delta \theta_{t_i} = \sum_{k=1}^{K} \left[ P(\tau_{i,k}) M_{t_i,k} \epsilon_{t_i,k} \right]
\]
- **Input**: DMP with initial parameters $\theta$, cost function $J$

- While (cost not converged)

  **Explore**
  - sample exploration vectors
  - execute DMP
  - determine cost

  **Update**
  - compute prob. from cost
  - prob.-weighted averaging
  - parameter update

\[
\frac{1}{\tau} \ddot{x}_t = \alpha (\beta (g - x_t) - \dot{x}_t) + g_t^T (\theta + \epsilon_{t,k})
\]

\[
J(\tau_i) = \phi_{tN} + \int_{t_i}^{tN} (q_t + \frac{1}{2} \theta_t^T R \theta_t) \, dt
\]

\[
\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma^\epsilon)
\]

\[
\theta_k = \theta + \epsilon_k
\]

\[
\theta \leftarrow \theta + \delta \theta
\]

\[
P(\tau_{i,k}) = \frac{e^{-\frac{1}{\lambda} J(\tau_{i,k})}}{\sum_{k=1}^{K} [e^{-\frac{1}{\lambda} J(\tau_{i,k})}]}
\]

\[
\delta \theta_{t_i} = \sum_{k=1}^{K} \left[ P(\tau_{i,k}) \, M_{t_i,k} \, \epsilon_{t_i,k} \right]
\]
Pi²- Algorithm

- Advantages
  - No gradient ⇒ Deals with discontinuous noisy cost functions
  - Update δ within convex hull of $\epsilon_{k=1...K}$ ⇒ Safe update rule
  - Arbitrary cost functions
  - Model-free
  - Fast convergence
  - Only one open parameter: magnitude of exploration ($\epsilon_{t,k} \sim \mathcal{N}(0, \Sigma)$)

- Disadvantage
  - No global convergence guarantees...
  - Robotics: This is where imitation comes in! $\pi(\theta^{imit}) \approx \pi(\theta^*)$

- Applied to very high-dimensional, complex tasks...
Cross-Entropy Method (CEM)

\[ N(\theta, \Sigma) \]
Cross-Entropy Method (CEM)

\[ \theta_{k=1 \ldots K} \sim \mathcal{N}(\theta, \Sigma) \]
Cross-Entropy Method (CEM)

\[ \theta_{k=1 \ldots K} \sim \mathcal{N}(\theta, \Sigma) \]
\[ \forall k \; J_k = J(\theta_k) \]
Cross-Entropy Method (CEM)

\[ \theta_{k=1...K} \sim \mathcal{N}(\theta, \Sigma) \]
\[ \forall k \ J_k = J(\theta_k) \]
\[ \theta_{k=1...K} \leftarrow \text{sort } \theta_{k=1...K} \text{ w.r.t } J_{k=1...K} \]

This algorithm can be interpreted as performing reward-weighted averaging.
Cross-Entropy Method (CEM)

\[ \theta_{k=1\ldots K} \sim \mathcal{N}(\theta, \Sigma) \]
\[ \forall k \quad J_k = J(\theta_k) \]
\[ \theta_{k=1\ldots K} \leftarrow \text{sort } \theta_{k=1\ldots K} \text{ w.r.t } J_{k=1\ldots K} \]
\[ \theta_{\text{new}} = \sum_{k=1}^{K_e} \frac{1}{K_e} \theta_k \]

This algorithm can be interpreted as performing reward-weighted averaging.
Cross-Entropy Method (CEM)

\[ \theta_{k=1 \ldots K} \sim \mathcal{N}(\theta, \Sigma) \]

\[ \forall k \ J_k = J(\theta_k) \]

\[ \theta_{k=1 \ldots K} \leftarrow \text{sort } \theta_{k=1 \ldots K} \text{ w.r.t } J_{k=1 \ldots K} \]

\[ \theta^{\text{new}} = \sum_{k=1}^{K_e} \frac{1}{K_e} \theta_k \]

\[ \Sigma^{\text{new}} = \sum_{k=1}^{K_e} \frac{1}{K_e} (\theta_k - \theta)(\theta_k - \theta)^\top \]
Cross-Entropy Method (CEM)

\[ \theta_{k=1\ldots K} \sim \mathcal{N}(\theta, \Sigma) \]
\[ \forall k \ J_k = J(\theta_k) \]
\[ \theta_{k=1\ldots K} \leftarrow \text{sort } \theta_{k=1\ldots K} \text{ w.r.t } J_{k=1\ldots K} \]
\[ \theta^{\text{new}} = \sum_{k=1}^{K_e} \frac{1}{K_e} \theta_k \]
\[ \Sigma^{\text{new}} = \sum_{k=1}^{K_e} \frac{1}{K_e} (\theta_k - \theta)(\theta_k - \theta)^T \]

This algorithm can be interpreted as performing reward-weighted averaging.
CMA-ES

- Covariance Matrix Adaptation - Evolutionary Strategy
- Like CEM, but
  - Different mapping from cost to probability
  - More sophisticated method for updating covariance matrix:

\[
p_\sigma \leftarrow (1 - c_\sigma) p_\sigma + \sqrt{c_\sigma (2 - c_\sigma) \mu_P \Sigma}^{-1} \frac{\theta_{\text{new}} - \theta}{\sigma}
\]  
\[\sigma_{\text{new}} = \sigma \times \exp \left( \frac{c_\sigma}{d_\sigma} \left( \frac{\|p_\sigma\|}{E\|\mathcal{N}(0, I)\|} - 1 \right) \right)\]  
\[
p_\Sigma \leftarrow (1 - c_\Sigma) p_\Sigma + h_\sigma \sqrt{c_\Sigma (2 - c_\Sigma) \mu_P} \frac{\theta_{\text{new}} - \theta}{\sigma}
\]  
\[
\Sigma_{\text{new}} = (1 - c_1 - c_\mu) \Sigma + c_1 (p_\Sigma p_\Sigma^T + \delta(h_\sigma) \Sigma)
\]
\[+ c_\mu \sum_{k=1}^{K_e} P_k (\theta_k - \theta)(\theta_k - \theta)^T\]
PI²/CEM/CMA-ES - Similarities

- PI²/CEM/CMA-ES are all based on
  - Exploration: sample from a Gaussian
  - Parameter update: Reward-Weighted Averaging

\[ \theta_k \sim \mathcal{N}(\theta, \Sigma) \]
\[ \theta^{new} = \sum_{k=1}^{K} P_k \theta_k \]
PL²/CEM/CMA-ES - Similarities

- PL²/CEM/CMA-ES are all based on
  - Exploration: sample from a Gaussian
  - Parameter update: Reward-Weighted Averaging

\[ \theta_k \sim \mathcal{N}(\theta, \Sigma) \]

\[ \theta^{new} = \sum_{k=1}^{K} P_k \theta_k \]

*Similarity striking, as algorithms derived from very different principles!*
\begin{itemize}
  \item \(\Pi^2/\text{CEM}/\text{CMA-ES}\) are all based on
    \begin{itemize}
      \item Exploration: sample from a Gaussian
      \item Parameter update: Reward-Weighted Averaging
    \end{itemize}

  \[
  \theta_k \sim \mathcal{N}(\theta, \Sigma) \\
  \theta^{new} = \sum_{k=1}^{K} P_k \theta_k
  \]

  \textit{Similarity striking, as algorithms derived from very different principles!}

  \item \(\text{CEM}\) is a special case of \(\text{CMA-ES}\): proof in paper.
    (maybe this was already known?)
\end{itemize}
\( \text{PI}^2/\text{CEM}/\text{CMA-ES} - \text{Differences} \)

- Also some differences
- Evaluated on following task:

\[
J(\tau_t) = \delta(t - 0.3) \cdot (x_t - 0.5)^2 + (y_t - 0.5)^2 + \sum_{D=1}^{D+1} ((D+1) - d) \ddot{a}_t^2 \sum_{D=1}^{D+1} ((D+1) - d)
\]
PI²/CEM/CMA-ES - Differences

- Also some differences
- Evaluated on following task:

\[ J(\tau_t) = \delta(t - 0.3) \cdot ((x_t - 0.5)^2 + (y_t - 0.5)^2) + \frac{\sum_{d=1}^{D}(D + 1 - d)(\ddot{a}_t)^2}{\sum_{d=1}^{D}(D + 1 - d)} \]
### Exploration Noise

<table>
<thead>
<tr>
<th>Method</th>
<th>Exploration Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>PI²</td>
<td>( \theta_{k,t} \sim \mathcal{N}(\theta, \Sigma) )</td>
</tr>
<tr>
<td>CEM</td>
<td>( \theta_k \sim \mathcal{N}(\theta, \Sigma) )</td>
</tr>
<tr>
<td>CMA-ES</td>
<td>( \theta_k \sim \mathcal{N}(\theta, \Sigma) )</td>
</tr>
</tbody>
</table>
### Exploration Noise

<table>
<thead>
<tr>
<th>Method</th>
<th>Noise Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{PI}^2 )</td>
<td>( \theta_{k,t} \sim \mathcal{N}(\theta, \Sigma) )</td>
</tr>
<tr>
<td>CEM</td>
<td>( \theta_k \sim \mathcal{N}(\theta, \Sigma) )</td>
</tr>
<tr>
<td>CMA-ES</td>
<td>( \theta_k \sim \mathcal{N}(\theta, \Sigma) )</td>
</tr>
</tbody>
</table>

![Comparison Diagram]

*Note: The diagram illustrates the comparison between different methods, showing the cost of evaluation trials over the number of trials.*
$PI^2$/CEM/CMA-ES - Differences

Eliteness
**PI²/CEM/CMA-ES - Differences**

**Eliteness**

**PI²**

**CEM**

**CMA-ES**

- **Comparison**
  - PI²
  - CMA-ES
- **Evaluation**
  - Task
  - Evaluation
### Covariance Matrix Updating

<table>
<thead>
<tr>
<th></th>
<th>PI²</th>
<th>CEM</th>
<th>CMA-ES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No</td>
<td>Yes (simple)</td>
<td>Yes (sophisticated)</td>
</tr>
</tbody>
</table>

**PI²/CEM/CMA-ES - Differences**

![Diagram](chart.png)
**PI^2-CMA**

- Suggests a new algorithm: PI^2-CMA
  - Constant exploration noise
  - Eliteness measure from PI^2
  - Covariance matrix updating from CEM/CMA-ES
PI$^2$-CMA

- Suggests a new algorithm: PI$^2$-CMA
  - Constant exploration noise
  - Eliteness measure from PI$^2$
  - Covariance matrix updating from CEM/CMA-ES

- Basically PI$^2$, but with adaptive exploration
  - In PI$^2$, exploration magnitude must be tuned by hand
  - Next evaluation demonstrates advantages of PI$^2$-CMA
$\Pi^2$-CMA- Adaptive Exploration

Learning Curve

Exploration Magnitude

$\log$ $\log$

$10^{2}$ $10^{3}$

number of trials

$10^{2}$ $10^{3}$ $10^{4}$

cost of evaluation trial

$10^{6}$ $10^{7}$ $10^{8}$

$\times 10^6$

linear
⇒ Exploration magnitude influences convergence speed and exploitation
PI\(^2\)-CMA- Adaptive Exploration

\[
\begin{align*}
\text{cost of evaluation trial} & \quad \text{exploration magnitude} \\
\lambda_{init} = 10^2 & \quad \lambda_{init} = 10^4 & \quad \lambda_{init} = 10^6 \\
\text{PI}^2 & \quad \text{PI}-CMA & \quad \text{PI}-CMA
\end{align*}
\]

number of trials

\(x \times 10^6\)

\(\lambda_{init} = 10^2\)

\(\lambda_{init} = 10^4\)

\(\lambda_{init} = 10^6\)

\(x20\)

Comparison

CMA-ES

Evaluation

Task

PI\(^2\)-CMA

Evaluation
\( \text{PI}^2 \)-CMA- Adaptive Exploration

![Graph showing comparison between PI2, CMA, CMA-ES, PI2-CMA, and PI2-CMA-ES. The graphs depict cost of evaluation trial and exploration magnitude against the number of trials.]

- PI\(^2\)
- CEM
- CMA-ES
- Comparison
- Evaluation
- Task
- PI\(^2\)-CMA

Legend:
- \( \lambda_{\text{init}} = 10^2 \)
- \( \lambda_{\text{init}} = 10^4 \)
- \( \lambda_{\text{init}} = 10^6 \)
PI²-CMA- Adaptive Exploration

More recent results (not in paper)
Pl$^2$-CMA- Adaptive Exploration

Comparison

CMA-ES

Evaluation

Task

Evaluation

Ball positioned 5cm lower at update 21

Trajectory cost of evaluation trial

degree of exploration

# updates
PI²/CEM/CMA-ES have identical update rules: reward-weighted averaging

Apply covariance matrix adaptation (as in CEM/CMA-ES) to PI²

Novel algorithm PI²-CMA
  - With adaptive exploration (other algorithmic parameters trivial to tune)

Future work
  - Further analysis and theoretical validation
  - Evaluation on real robots
Thank you for your attention!

Questions?
Hierarchical reinforcement learning with motion primitives.
In 11th IEEE-RAS International Conference on Humanoid Robots.

Learning to grasp under uncertainty.

Learning motion primitive goals for robust manipulation.
In International Conference on Intelligent Robots and Systems (IROS).