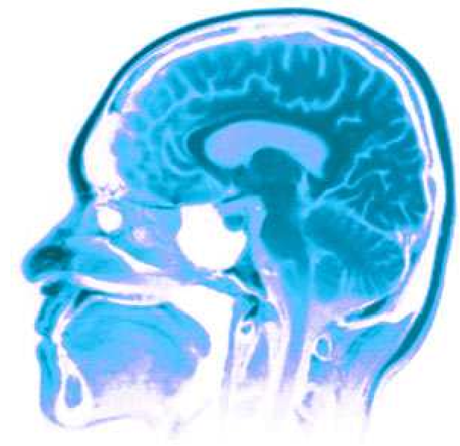
CPSC540

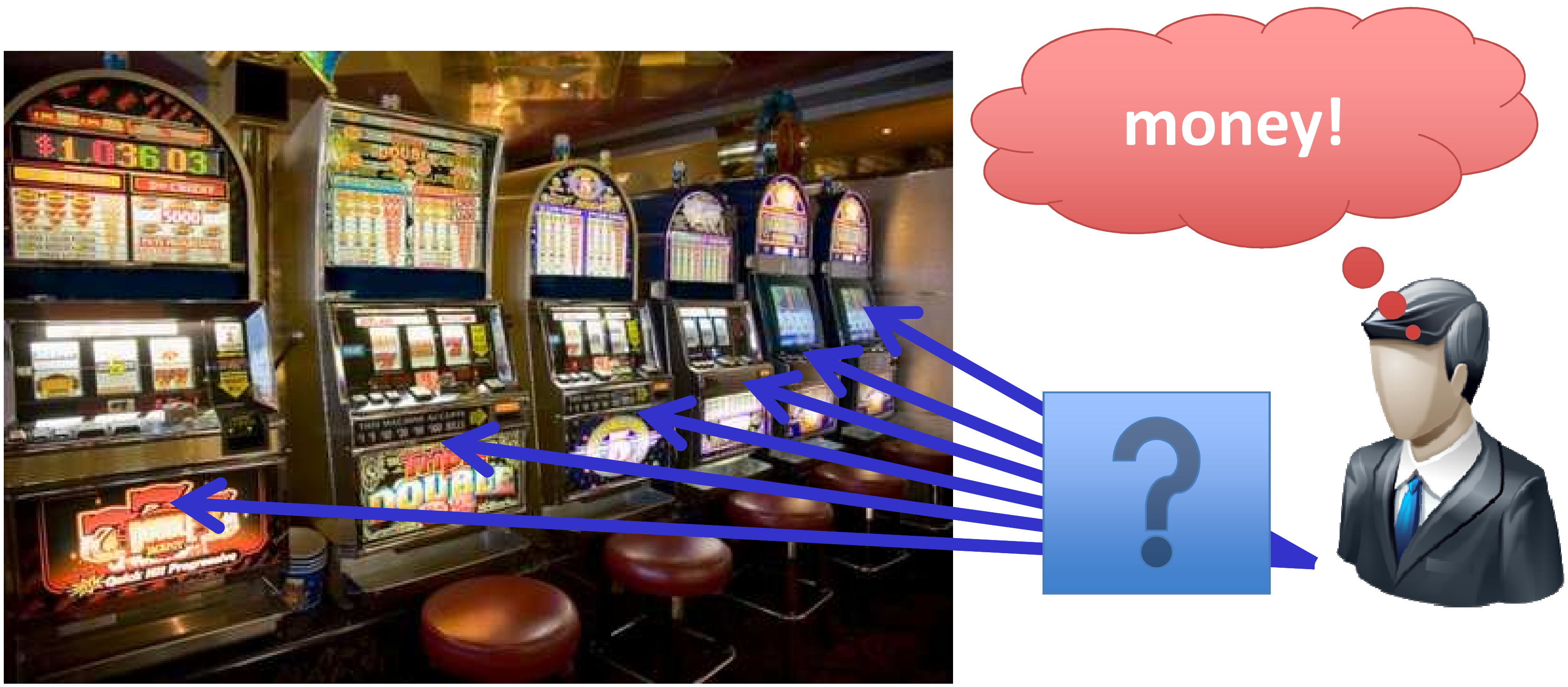


Bayesian optimization, bandits and Thompson sampling

Nando de Freitas

February 2013

Multi-armed bandit problem



Multi-armed bandit problem

Actions

$\{1, 2, \ldots, K\}$

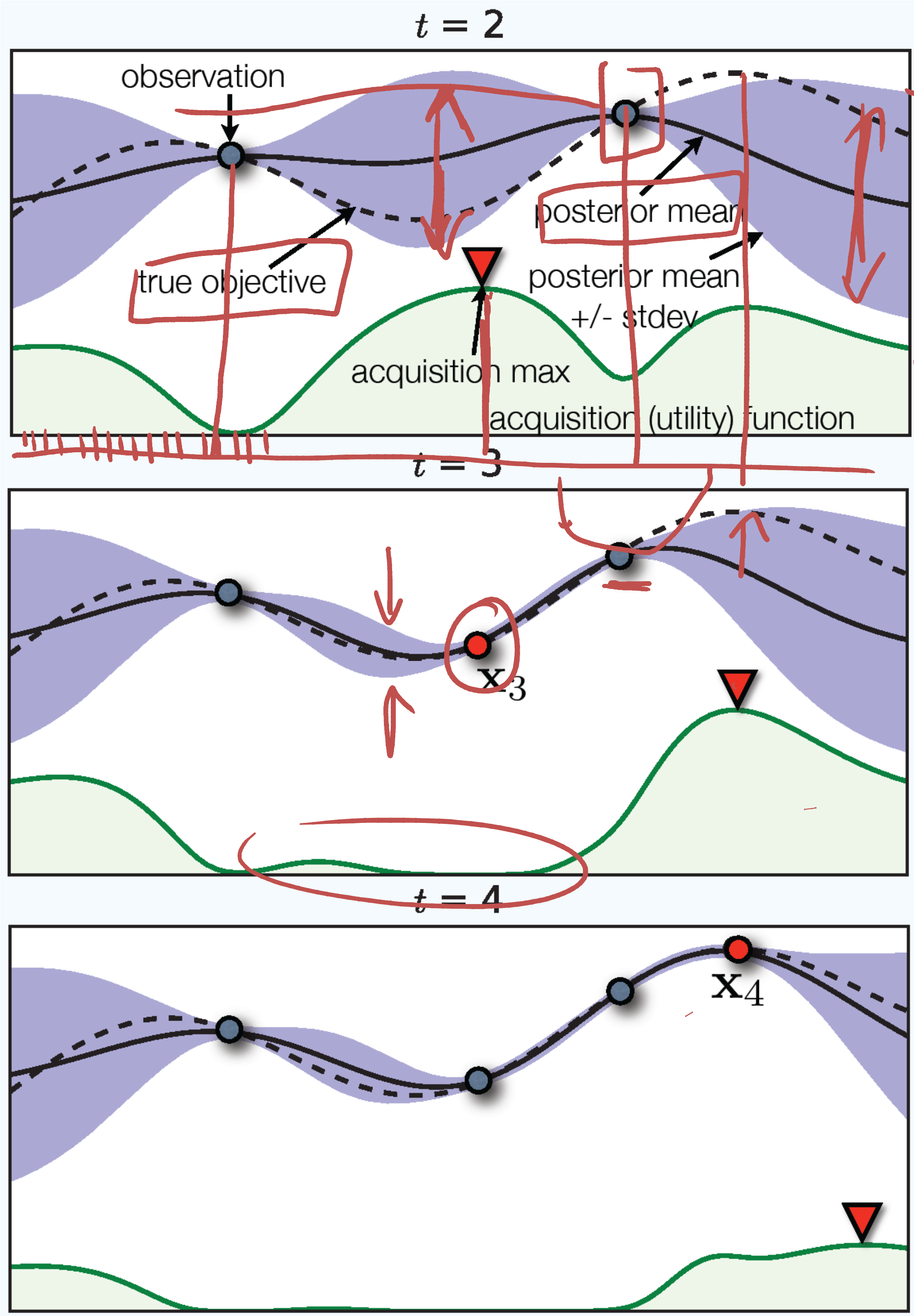
Reward(s)

$\mathbf{x}(t) \in [0,1]^K$

$t = 1, 2, \ldots, T$

Sequence of trials

- **Trade-off between Exploration and Exploitation**
- **Regret = Player reward – Reward of best action**



$\{(x_1, y_1)\ (x_2, y_2)\}$

$u(x \mid \text{data})$

Bayesian optimization

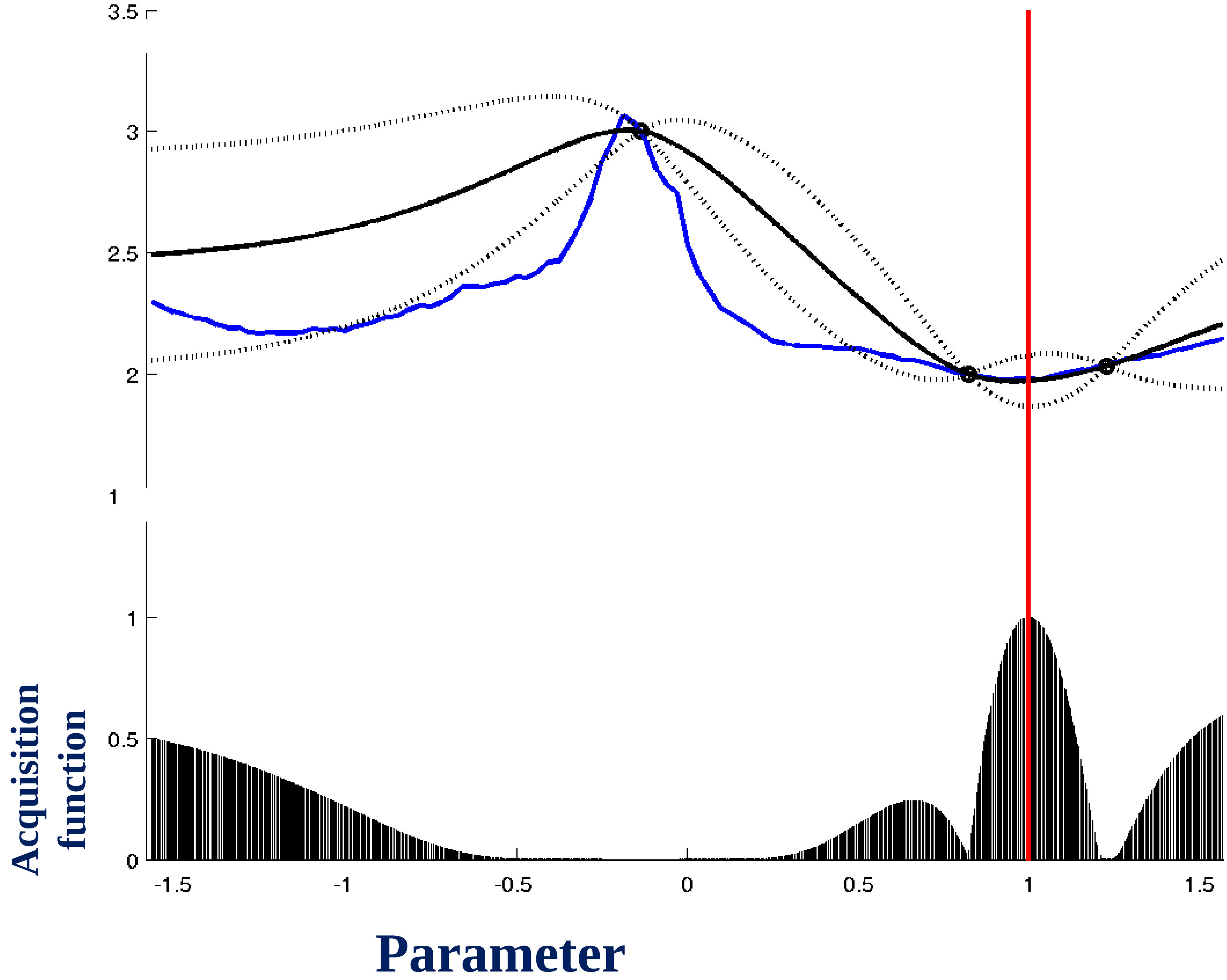
1: **for** $t = 1, 2, \ldots$ **do**

2: Find $\mathbf{x}_t$ by combining attributes of the posterior distribution in a utility function u and maximizing:
$\mathbf{x}_t = \operatorname{argmax}_{\mathbf{x}} u(\mathbf{x} \mid \mathcal{D}_{1:t-1})$.

3: Sample the objective function:
$y_t = f(\mathbf{x}_t) + \varepsilon_t$.

4: Augment the data $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t)\}$ and update the GP.

5: **end for**



3.5
3
2.5
2
1
1
0.5
0
-1.5
-1
-0.5
0
0.5
1
1.5
Acquisition function
Parameter

Exploration-exploitation tradeoff

Recall the expressions for GP prediction:

$$P(\underbrace{y_{t+1}}_{\text{Prediction}}|\mathcal{D}_{1:t}, \underbrace{\mathbf{x}_{t+1}}_{\text{test}}) = \mathcal{N}(\mu_t(\mathbf{x}_{t+1}), \sigma_t^2(\mathbf{x}_{t+1}) + \sigma^2_{\text{noise}})$$

$$\mu_t(\mathbf{x}_{t+1}) = \mathbf{k}^T[\mathbf{K} + \sigma^2_{\text{noise}}\mathbf{I}]^{-1}\mathbf{y}_{1:t}$$

$$\sigma_t^2(\mathbf{x}_{t+1}) = k(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}) - \mathbf{k}^T[\mathbf{K} + \sigma^2_{\text{noise}}\mathbf{I}]^{-1}\mathbf{k}$$

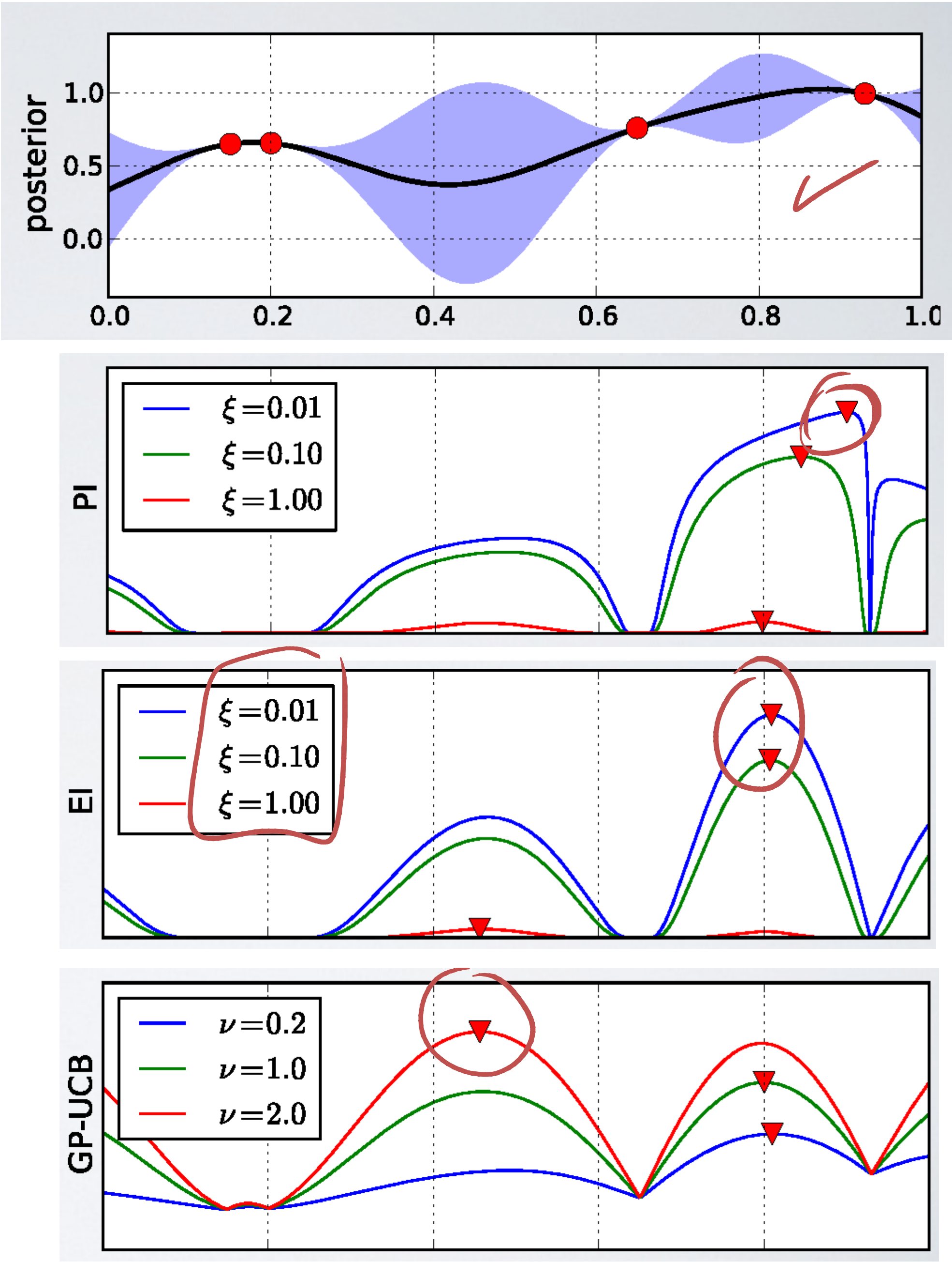
We should choose the next point ***x*** where the mean is high (exploitation) and the variance is high (exploration).

We could balance this tradeoff with an acquisition function as follows:

$$\mu(\mathbf{x}) + \kappa\sigma(\mathbf{x})$$

Acquisition functions

- aka infill, figure of merit
- acquisition function guides the optimization by determining which $\mathbf{x}_{t+1}$ to observe next
- uses predictive posterior to combine exploration (high-variance regions) and exploitation (high-mean regions)
- optimize to find sample point (can be done cheaply/approximately)



An acquisition function: Probability of Improvement

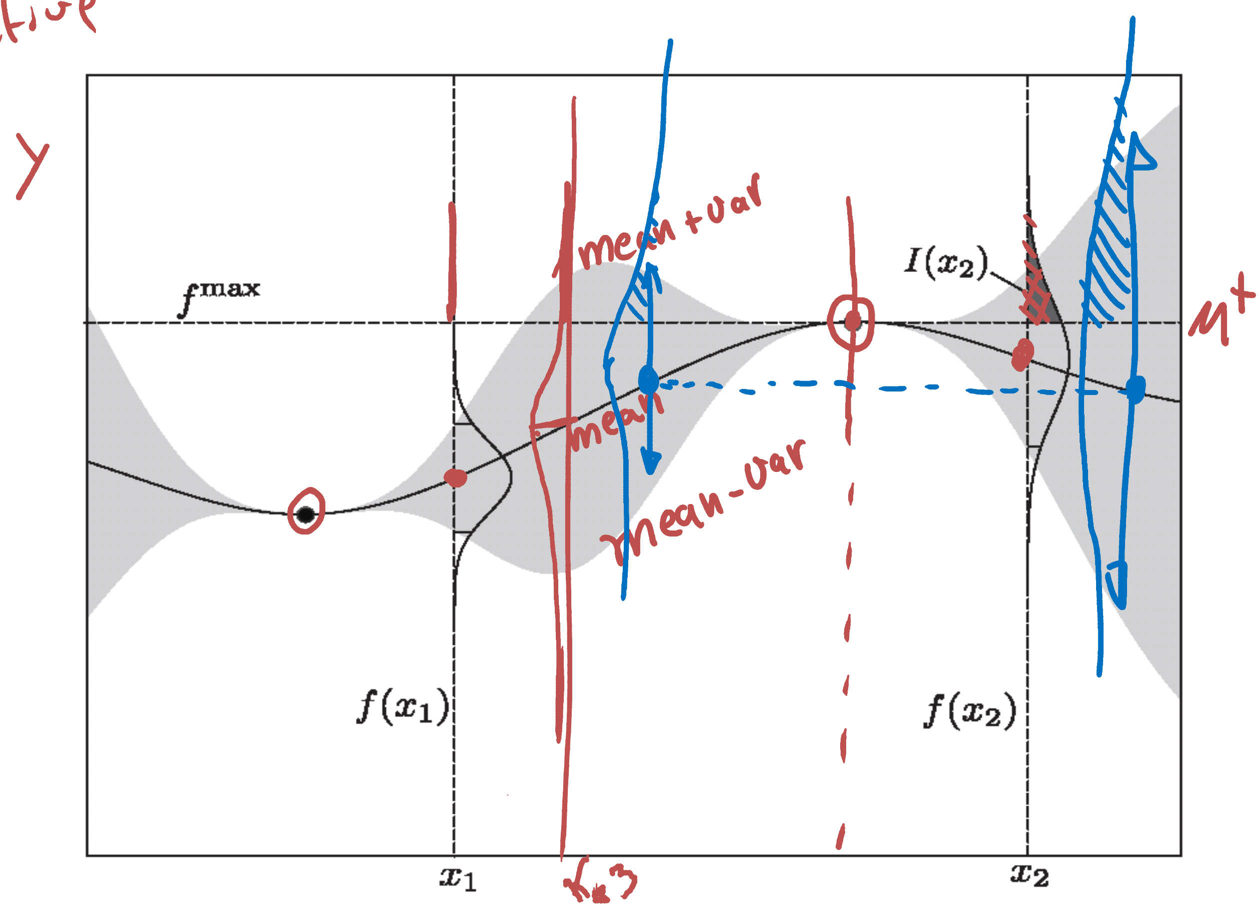
$$\mathrm{PI}(\mathbf{x}) = P(f(\mathbf{x}) \geq \mu^{+} + \cancel{\xi}\,0)$$

$$= \Phi\left(\frac{\mu(\mathbf{x}) - \mu^{+} - \xi}{\sigma(\mathbf{x})}\right)$$

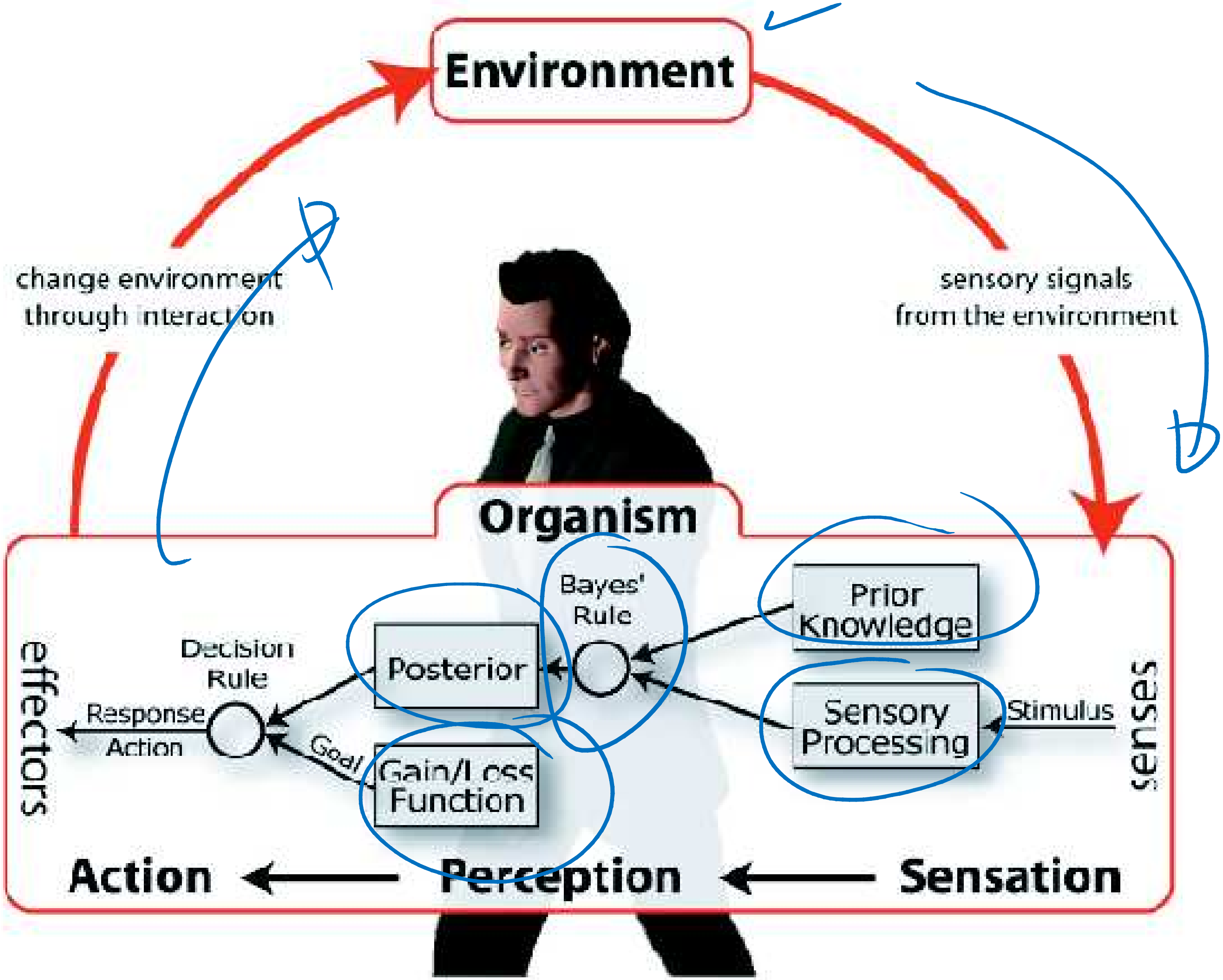
cumulative

μ^+ best observed value

$f^{max} = \mu^+ + \xi$ small



People as Bayesian reasoners



Bayes and decision theory

Utilitarian view: We need models to make the right decisions under uncertainty. Inference and decision making are intertwined.

Learned posterior

$$\begin{cases} P(x=\text{healthy}|\text{data}) = 0.9 \; ✓ \\ P(x=\text{cancer}|\text{data}) = 0.1 \; ✓ \end{cases}$$

Cost/Reward model *u(x,a)*

	$\mathbf{a} = no\ treatment$	$\mathbf{a} = treatment$
$\mathbf{x} = healthy$	0	-30
$\mathbf{x} = cancer$	-100	-20

We choose the action that maximizes the expected utility, or equivalently, which minimizes the expected cost.

$$EU(a) = \sum_x u(x,a)\, P(x|data)$$

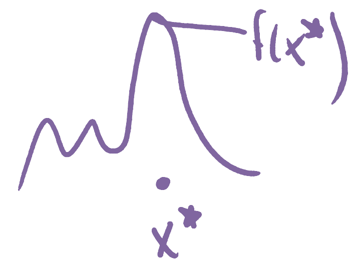
$EU(a=treatment) =$ u(healthy, treatment) P(x=healthy|d) + u(cancer, treatmt.) P(cancer|d)

(-30)(0.9) + (-20)(0.1) =

$EU(a=no\ treatment) =$

An expected utility criterion

At iteration *n+1*, choose the point that minimizes the distance to the objective evaluated at the maximum *x**:



GP

true f

$$\mathbf{x}_{n+1} = \arg\min_{\mathbf{x}} \mathbb{E}(\|f_{n+1}(\mathbf{x}) - f(\mathbf{x}^\star)\| \,|\mathcal{D}_n)$$
$$= \arg\min_{\mathbf{x}} \int \|f_{n+1}(\mathbf{x}) - f(\mathbf{x}^\star)\| p(f_{n+1}|\mathcal{D}_n) df_{n+1}$$

We don’t know the true objective at the maximum. To overcome this, Mockus proposed the following acquisition function:

$$\mathbf{x} = \arg\max_{\mathbf{x}} \mathbb{E}(\max\{0, f_{n+1}(\mathbf{x}) - f^{\max}\} \,|\mathcal{D}_n)$$

Expected improvement

$$\mathbf{x} = \arg\max_{\mathbf{x}} \mathbb{E}(\max\{0, f_{n+1}(\mathbf{x}) - \underbrace{f^{\max}}_{\mu^+ + \xi}\} \,|\mathcal{D}_n)$$

For this acquisition, we can obtain an analytical expression:

$$\mathrm{EI}(\mathbf{x}) = \begin{cases} (\mu(\mathbf{x}) - \mu^+ - \xi)\Phi(Z) + \sigma(\mathbf{x})\phi(Z) & \text{if } \sigma(\mathbf{x}) > 0 \\ 0 & \text{if } \sigma(\mathbf{x}) = 0 \end{cases}$$

$$Z = \frac{\mu(\mathbf{x}) - \mu^+ - \xi}{\sigma(\mathbf{x})}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ denote the PDF and CDF of the standard Normal

A third criterion: GP-UCB

Define the *regret* and cumulative regret as follows:

$$\begin{aligned} r(\mathbf{x}) &= f(\mathbf{x}^\star) - f(\mathbf{x}) \\ R_T &= r(\mathbf{x}_1) + \cdots + r(\mathbf{x}_T) \end{aligned}$$

The GP-UCB criterion is as follows:

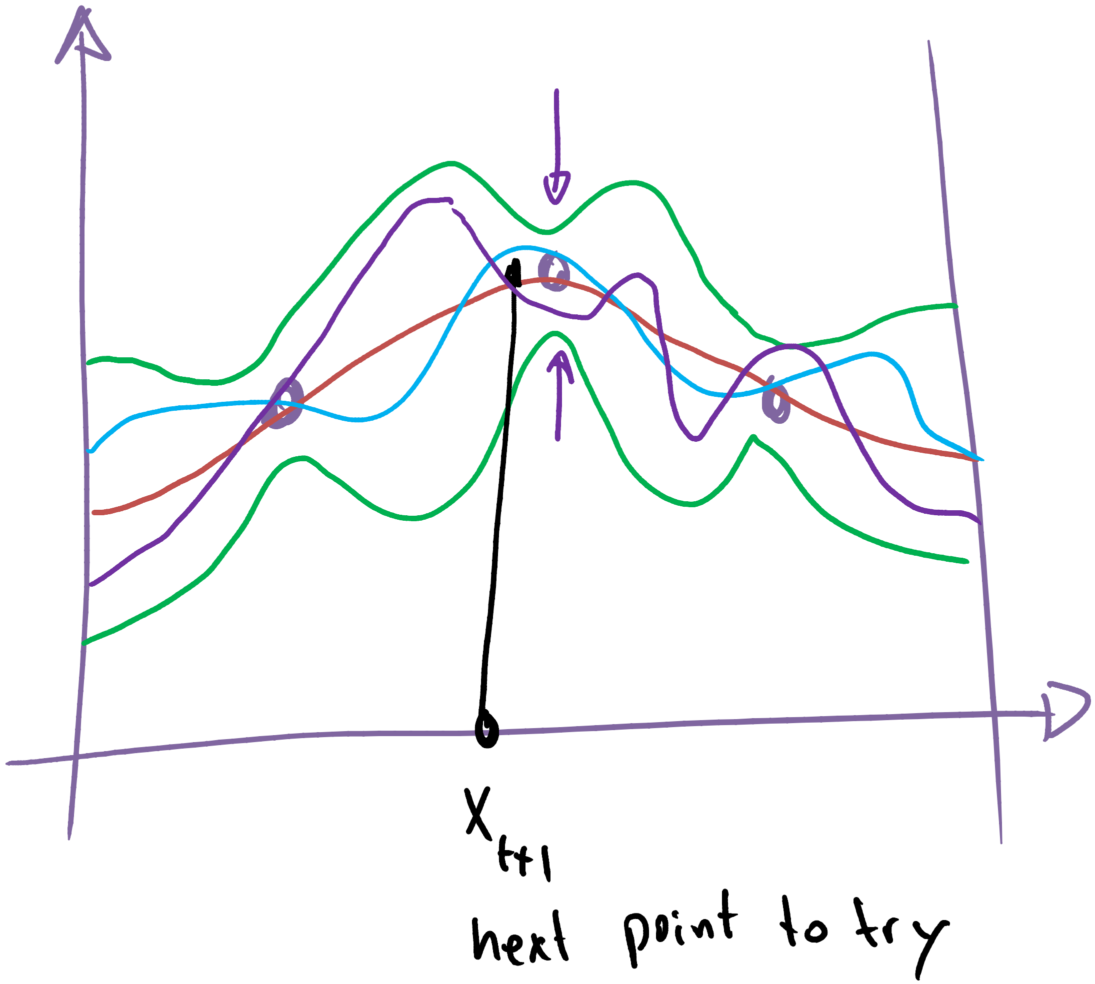
$$\text{GP-UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \sqrt{\nu\beta_t}\sigma(\mathbf{x})$$

Beta is set using a simple concentration bound:

With $\nu = 1$ and $\beta_t = 2\log(t^{d/2+2}\pi^2/3\delta)$, it can be shown[2] with high probability that this method is *no regret*, i.e. $\lim_{T\to\infty} R_T/T = 0$. This in turn implies a lower-bound on the convergence rate for the optimization problem.

[Srinivas et al, 2010]

A fourth criterion: Thompson sampling



Acquisition functions

$$\boxed{\mu^+ = \mathrm{argmax}_{\mathbf{x}_i \in \mathbf{x}_{1:t}} \mu(\mathbf{x}_i)}$$

- Probability of Improvement

$$\mathrm{PI}(\mathbf{x}) = \Phi\left(\frac{\mu(\mathbf{x}) - \mu^+ - \xi}{\sigma(\mathbf{x})}\right)$$

Kushner 1964

- Expected Improvement

$$\mathrm{EI}(\mathbf{x}) = (\mu(\mathbf{x}) - \mu^+ - \xi)\Phi(Z) + \sigma(\mathbf{x})\phi(Z)$$

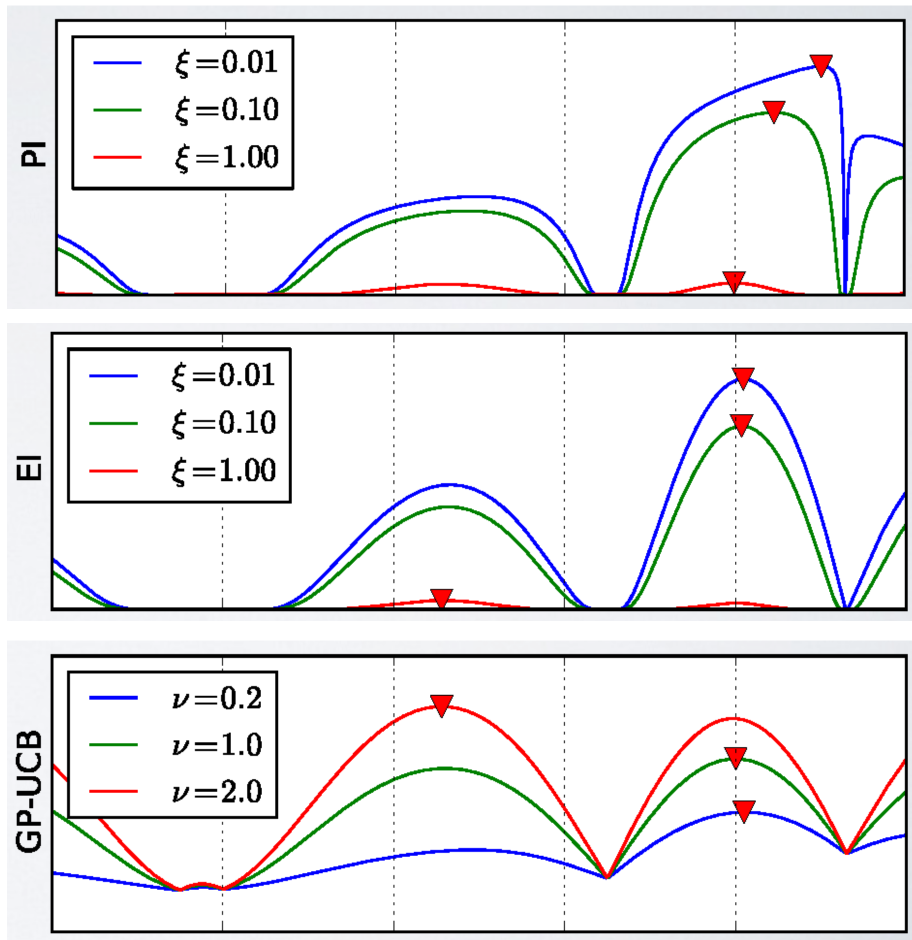
$$Z = \frac{\mu(\mathbf{x}) - \mu^+ - \xi}{\sigma(\mathbf{x})}$$

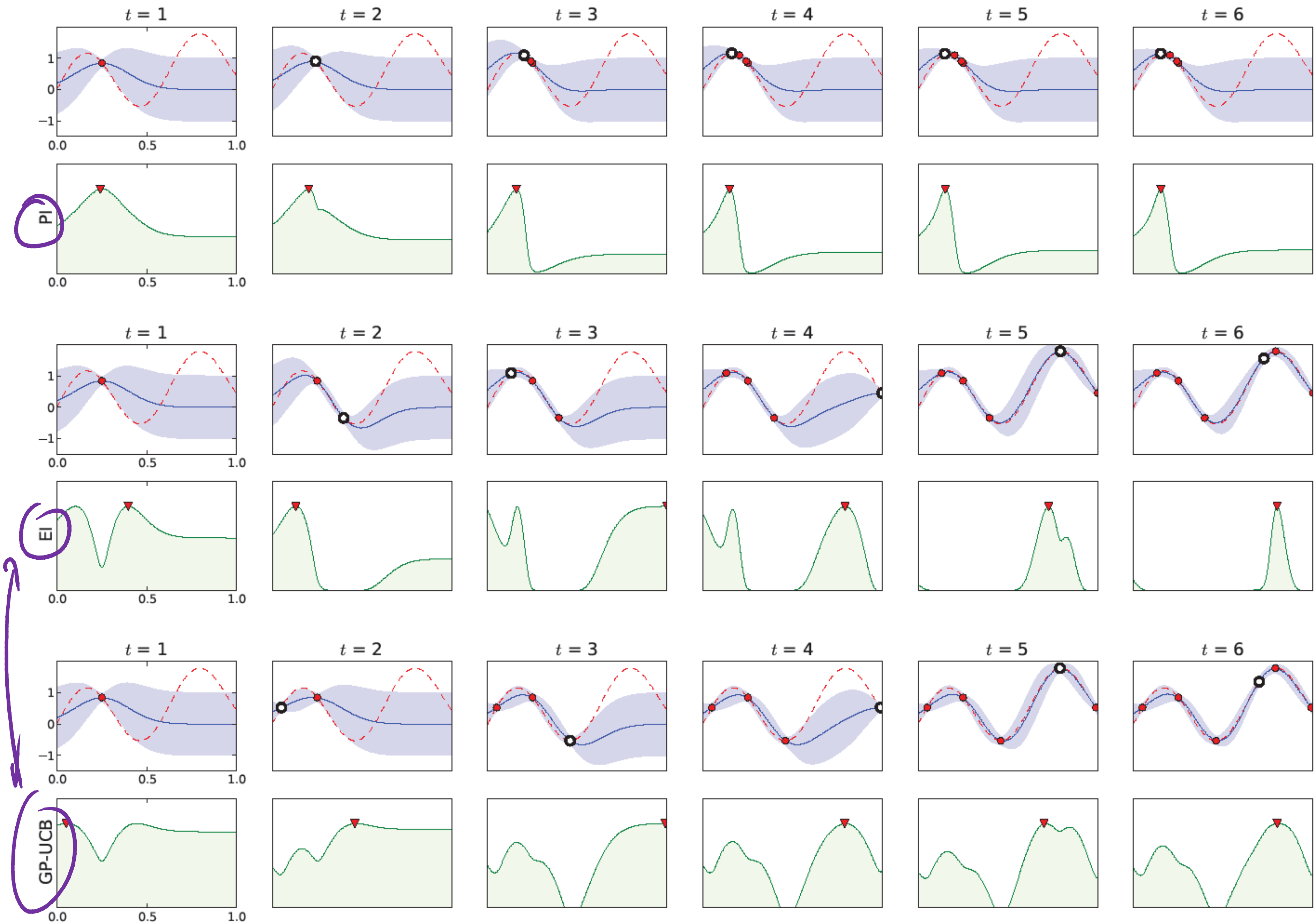
Mockus 1978

- Upper Confidence Bound

$$\text{GP-UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \sqrt{\nu\tau_t}\sigma(\mathbf{x})$$

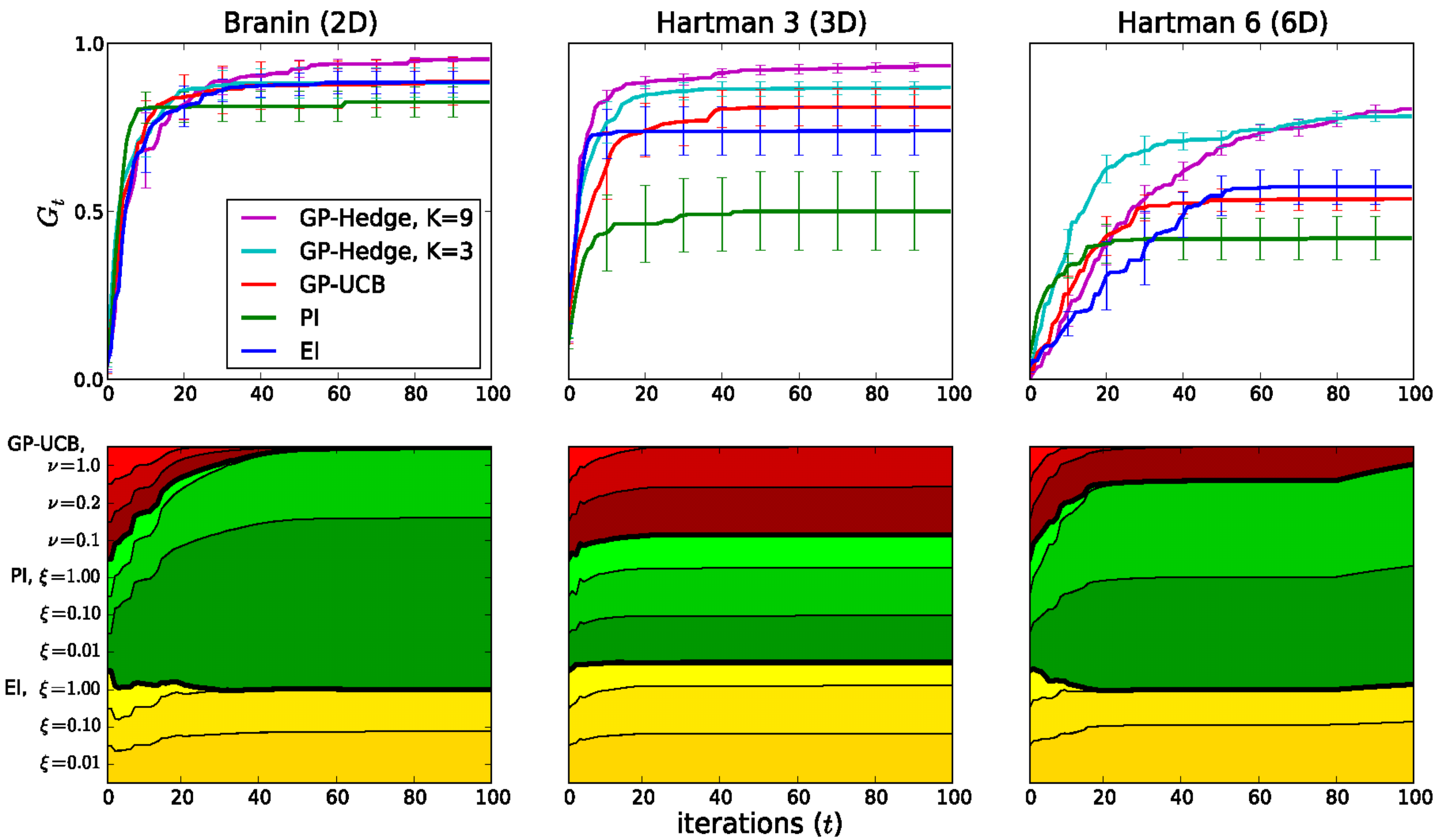
Srinivas *et al.* 2010



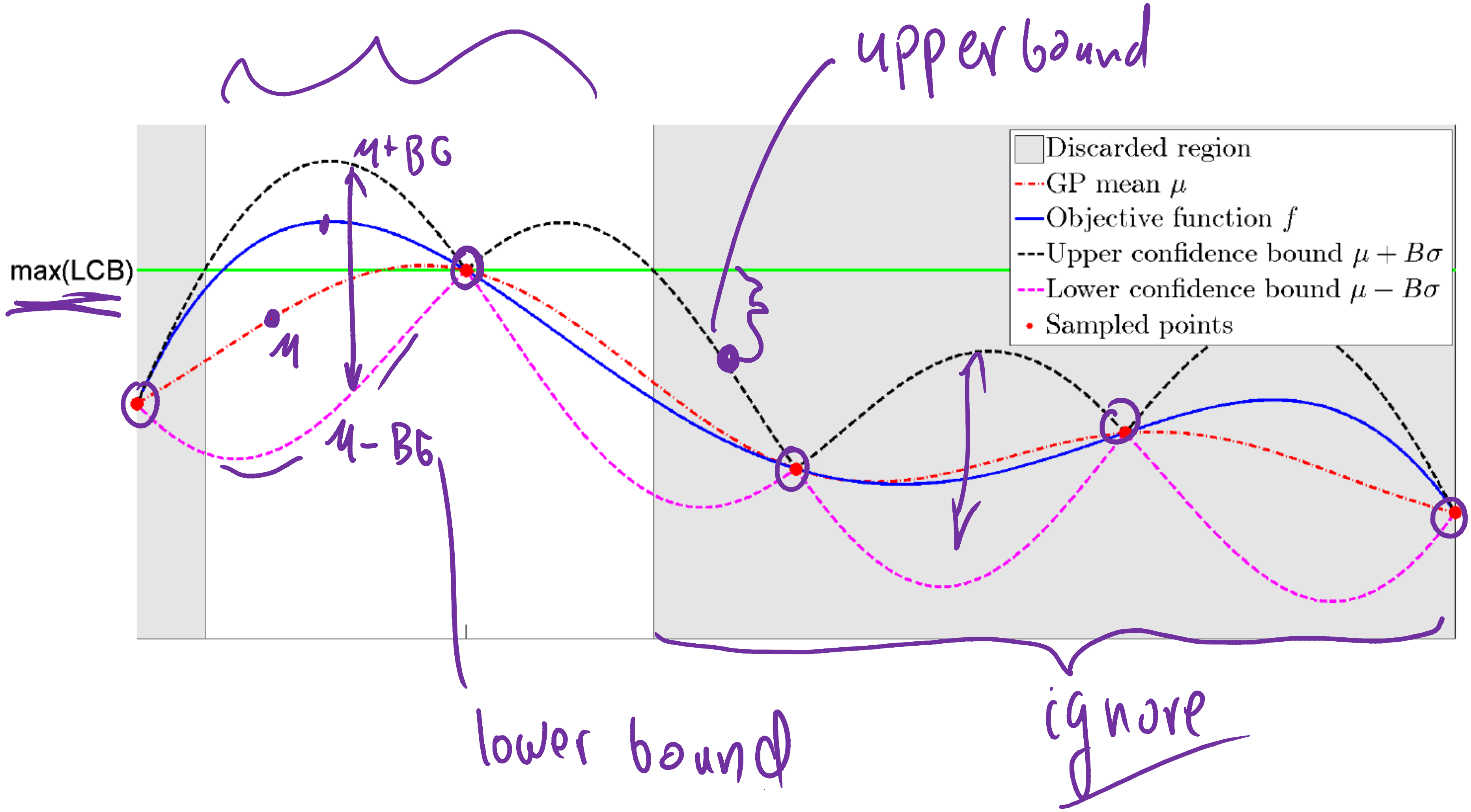


t = 1
t = 2
t = 3
t = 4
t = 5
t = 6
PI
EI
GP-UCB
1
0
−1
0.0
0.5
1.0

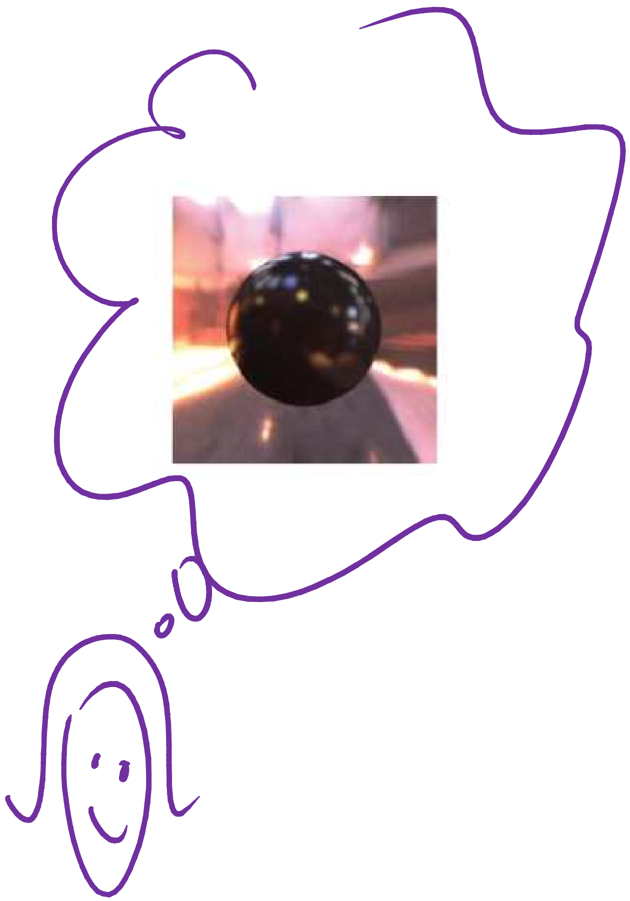
Portfolios of acquisition functions help



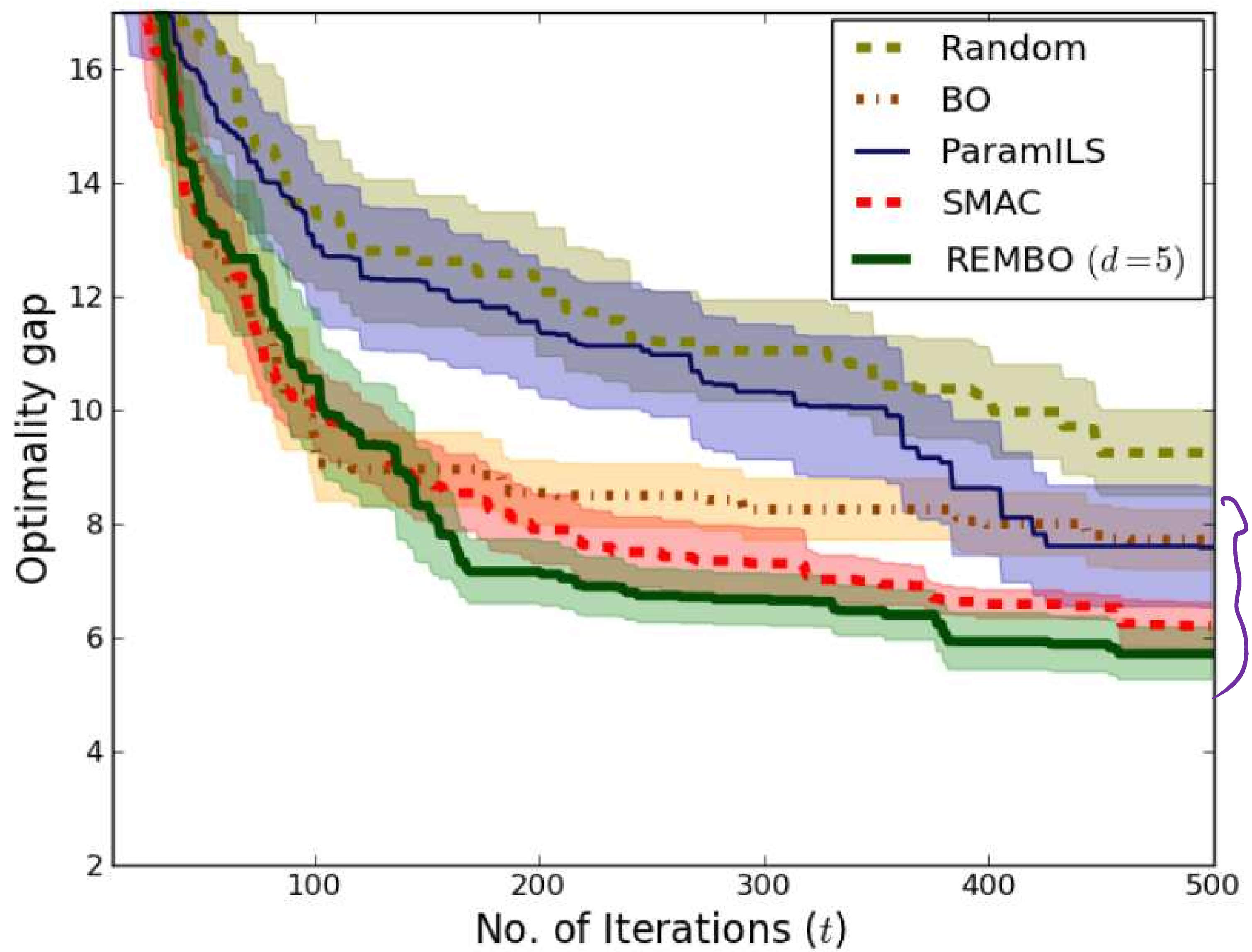
Why Bayesian Optimization works



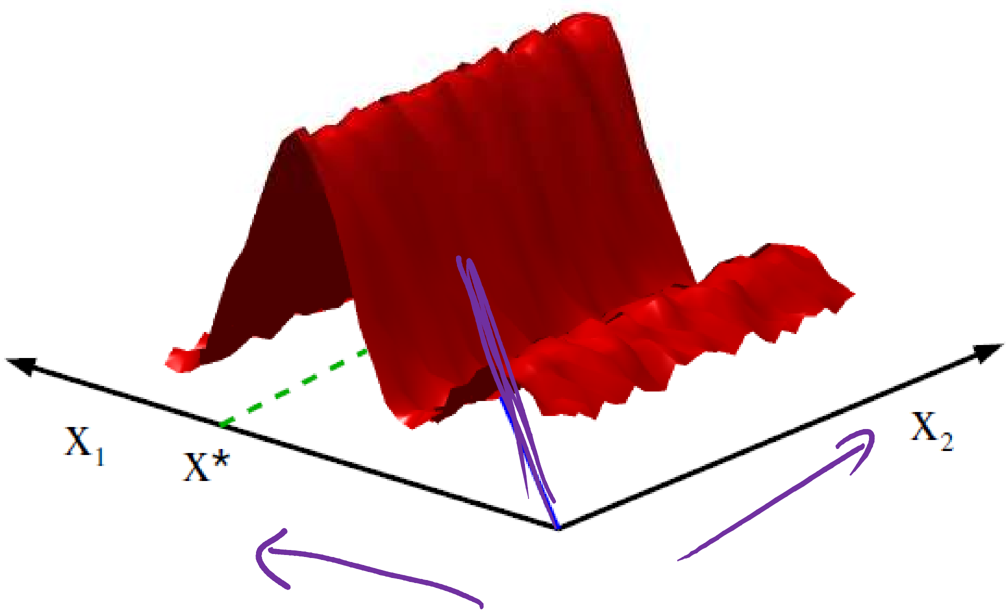
Intelligent user interfaces

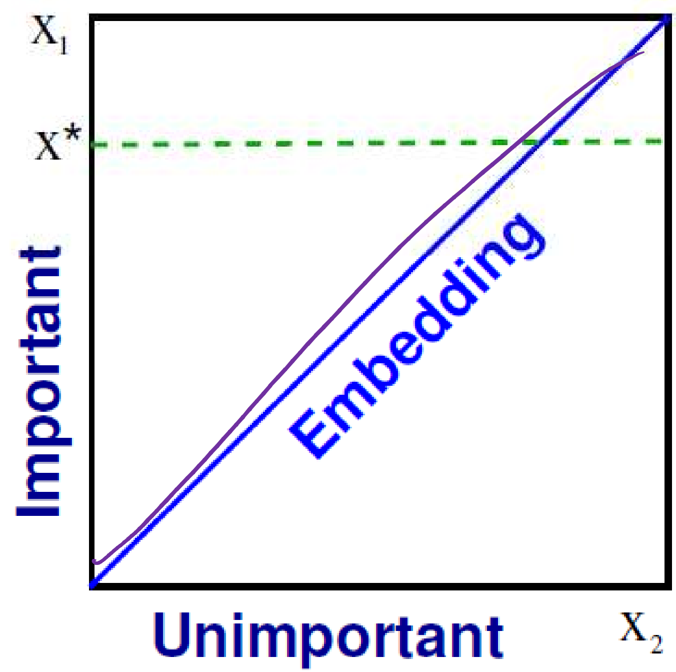


Example: Tuning NP hard problem solvers

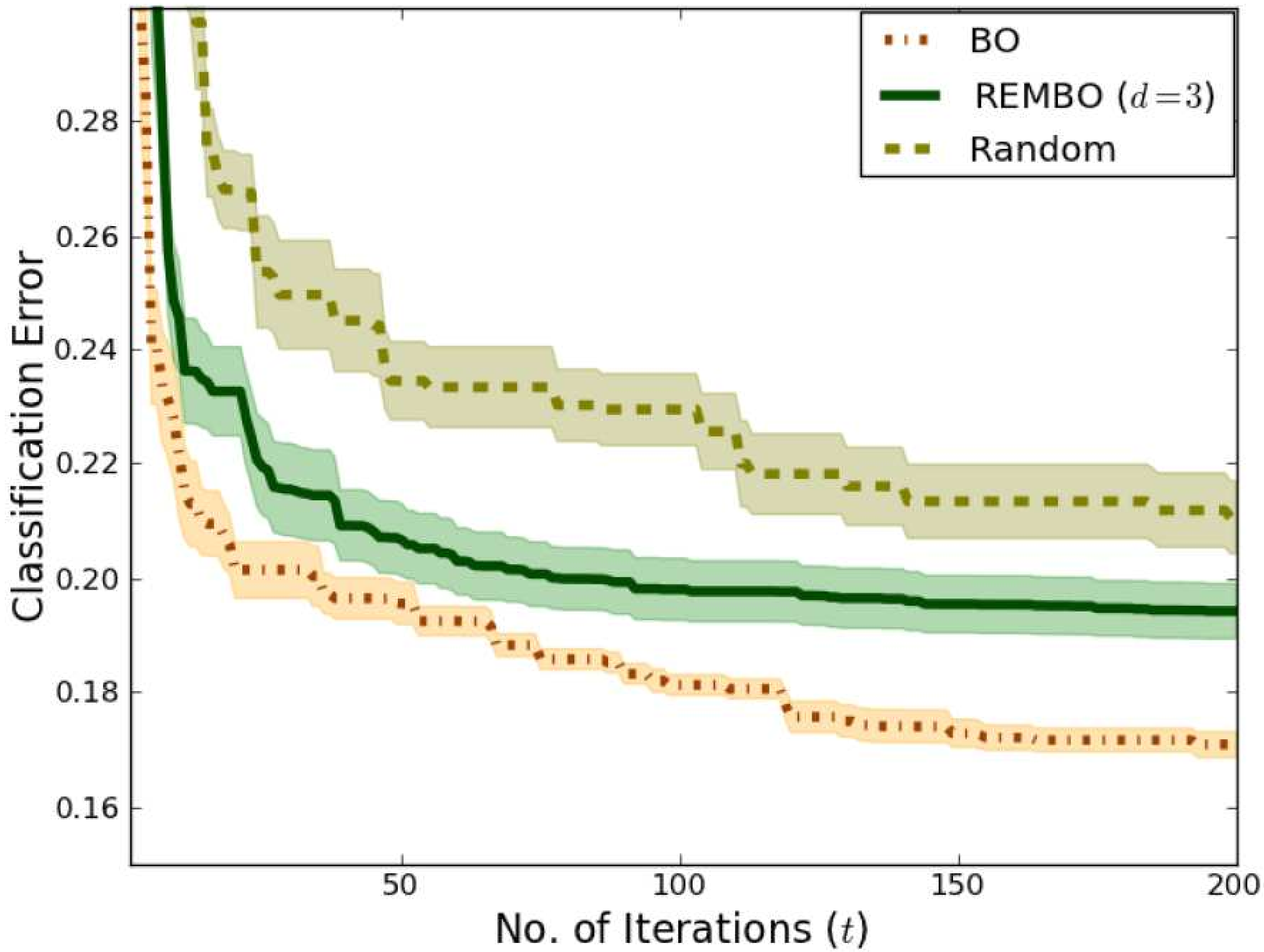


Why random tuning works sometimes





Example: Tuning random forests



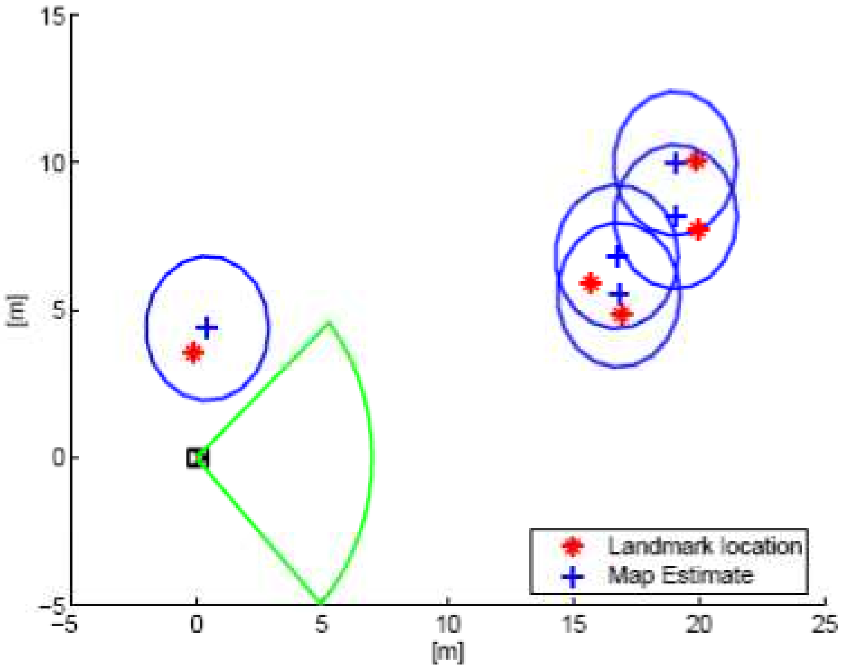
Example: Tuning hybrid Monte Carlo

Table 4.1: Mean squared test error for the robot arm data set.

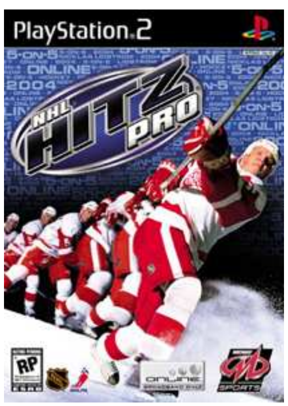
Method	**Mean Squared Error**
Rios Insua and Muller's (1998) MLP with reversible-jump MCMC	0.00620
Mackay's (1992) Gaussian approximation with highest evidence	0.00573
Neal's (1996) HMC	0.00554
Neal's (1996) HMC with ARD	0.00549
Reversible-jump MCMC with Bayesian model by Andrieu et al.	0.00502
Adaptive HMC (Median Error)	0.00499
Adaptive HMC (Mean Error)	0.00498 ± 0.00012

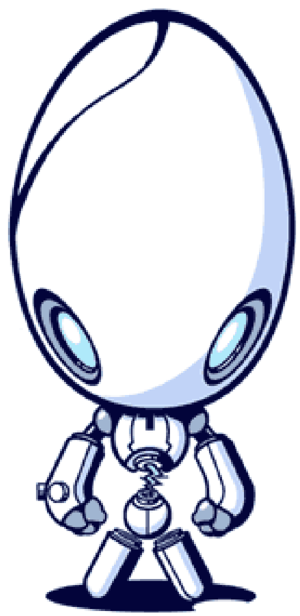
Table 4.2: Classification error on the validation set of the Dexter data set.

Method	**Classification Error**
New-Bayes-nn-sel	0.0800
Adaptive HMC (Mean error)	0.0730 ± 0.0096
Adaptive HMC (Median error)	0.0700
Adaptive HMC + Majority Voting	0.0667


15
10
5
0
−5
[m]
−5
0
5
10
15
20
25
[m]
Landmark location
Map Estimate

The games industry, rich in sophisticated large-scale simulators, is a great environment for the design and study of automatic decision making systems.



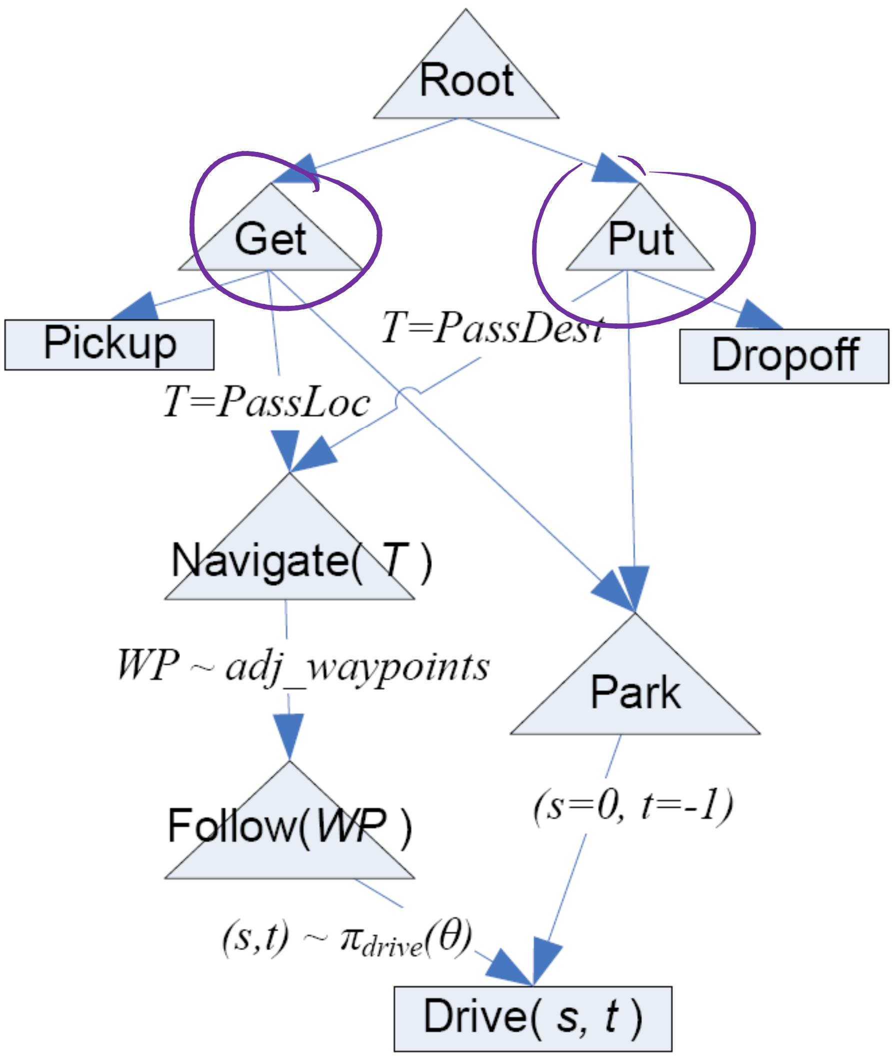


Hierarchical policy example

– High-level model-based learning for deciding when to navigate, park, pickup and dropoff passengers.

– Mid-level active path learning fo navigating a topological map.

– Low-level active policy optimizer to learn control of continuous non-linear vehicle dynamics.

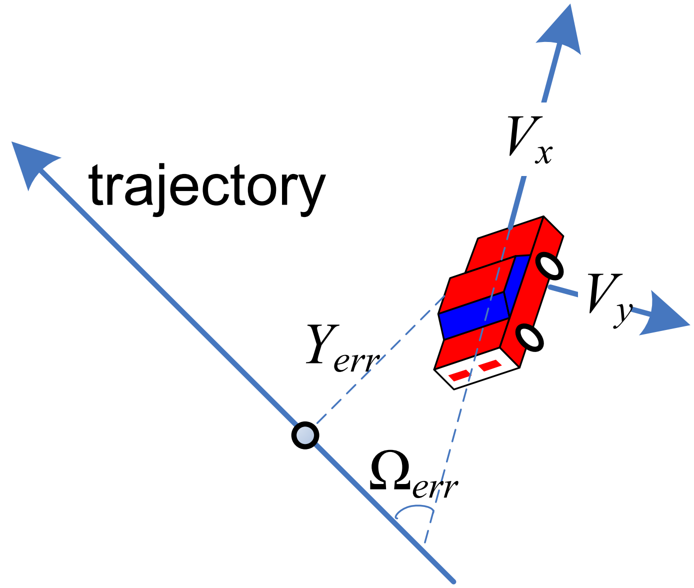


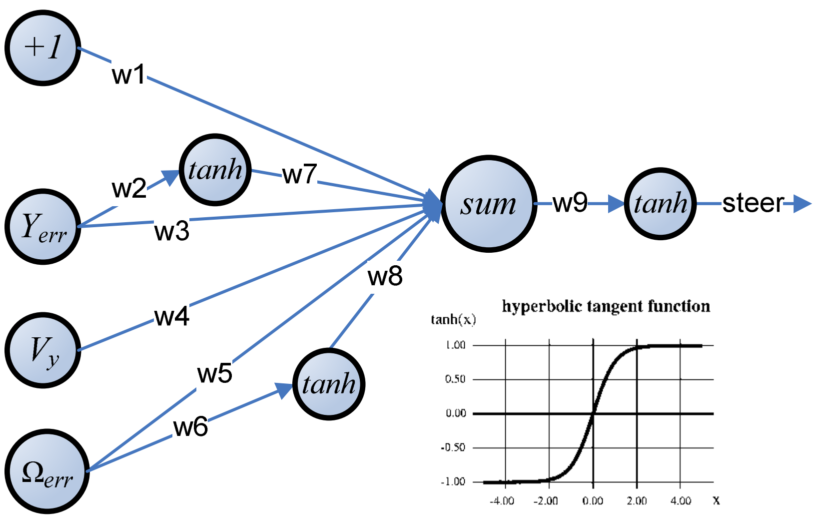
Active Path Finding in Middle Level

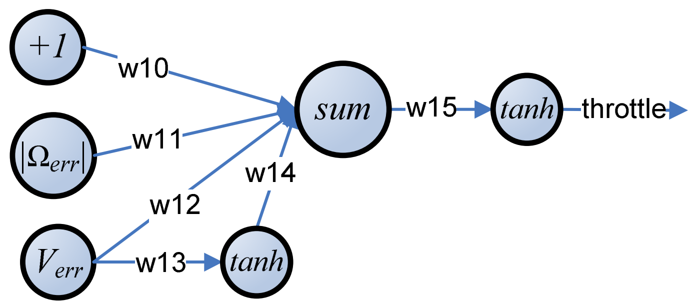
- Mid-level *Navigate* policy generates sequence of waypoints on a topological map to navigate from a location to a destination. *V(θ)* value function represents the path length from the current node, to the target.

Low-Level: Trajectory following

TORCS: 3D game engine that implements complex vehicle dynamics complete with manual and automatic transmission, engine, clutch, tire, and suspension models.





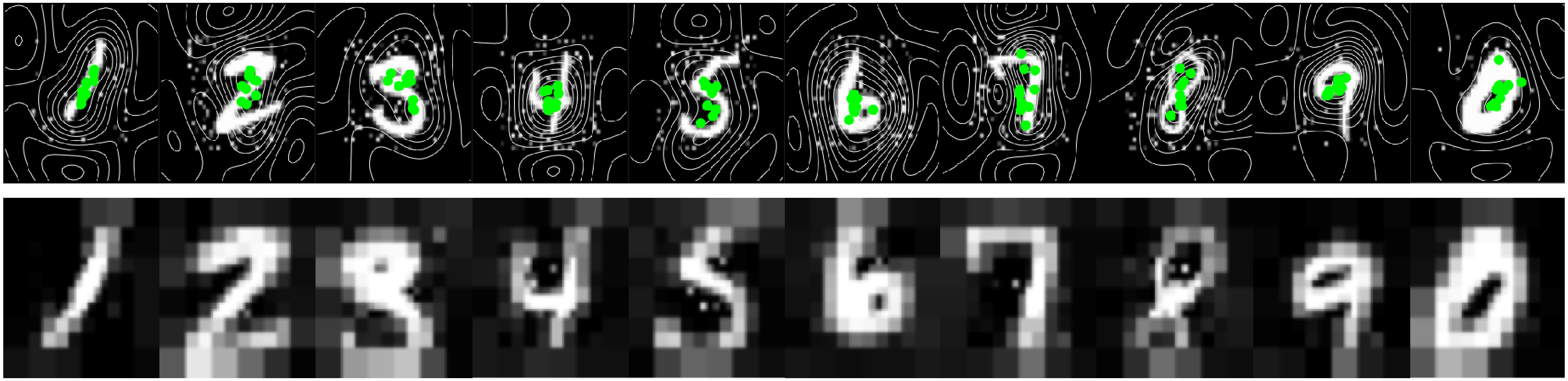


Hierarchical systems apply to many robot tasks – key to build large systems

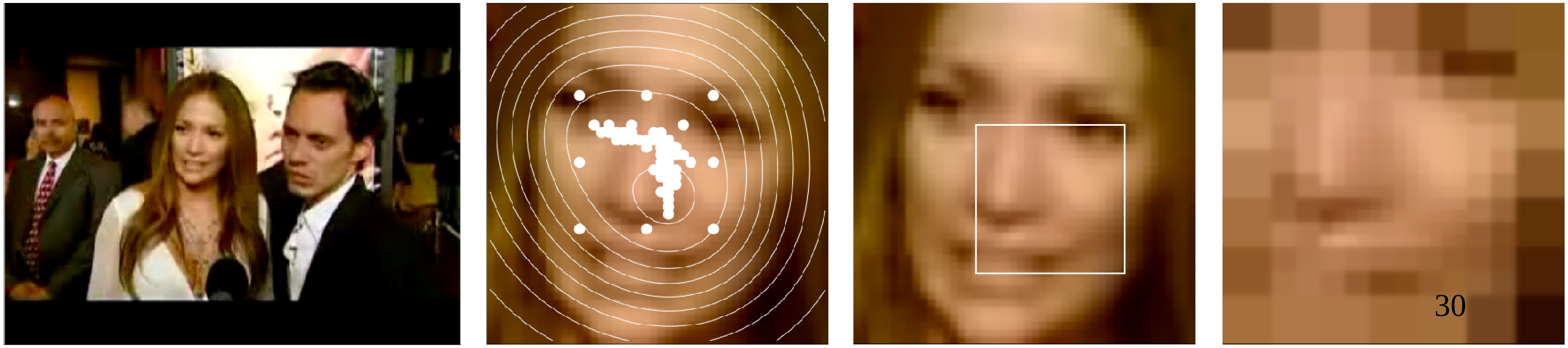
We used TORCS: A 3D game engine that implements complex vehicle dynamics complete with manual and automatic transmission, engine, clutch, tire, and suspension models.

Gaze planning

Digits Experiment:



Face Experiment:



Next lecture

In the next lecture, we embark on our quest to learn all about random forests. We will begin by learning about decision trees.