Introduction to Reinforcement Learning

Arvind Kumar

Computational Brain Science
Computational Science & Technology
KTH, Royal Institute of Technology, Stockholm, Sweden

CAMP @ NCBS – July 3 2016

Learning

- **Supervised learning**

  Categarization of inputs based on the output ? no role of the internal structure of the inputs – formation of input out association based on the error signal

- **Unsupervised learning**

  Categorisation of input based in their internal structure

- **Reinforcement learning**

  Agent is only told if it has performed the right or wrong action

  RL is more natural - often teacher who knows all is not available

  and even when such a teacher is around, there is no guarantee that the learner can interpret the teacher correctly

The Law of Effect

Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur;
those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur.
The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond. (Thorndike, 1911, p. 244)

Markov decision process

The response of a system to an action taken at time t is described by the probability

$$Pr\{R_{t+1} = r, S_{t+1} = s' | S_0, A_0, R_1, S_1, A_1, \dots R_{t-1}, S_{t-1}, A_{t-1}, R_t, S_t, A_t\}$$

When the system as a *Markov Property*

$$Pr\{R_{t+1} = r, S_{t+1} = s' | S_t, A_t\}$$

If this holds, we can predict the next state and reward given the current state and action. And all future rewards iteratively.

Markov decision process

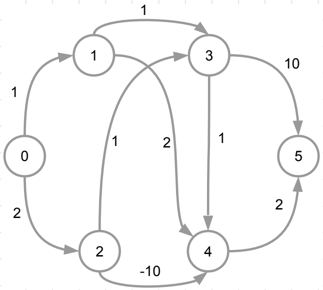
A RL task in an environment with Markov property is called the *Markov decision process*.

And the state transition probability is

$$p(s'|s,a) = Pr\{s_{t+1} = s'|s_t = s, a_t = a\}$$

The expected reward is

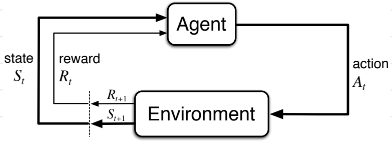
$$r(s,a,s') = \mathbb{E}[r_{t+1}|s_t = s, s_t = a, s_{t+1} = s']$$



Reinforcement learning problem

- $s_t \in \mathcal{S}$ state at time t
- $a_t, \in \mathcal{A}(S_t)$ action available in the state S_t
- $r_{t+1} \in \mathbb{R}$ numerical reward
- s_{t+1} is the next state after an action a was taken in the state s_t
- π is the Policy that maps the state s to an action a

  $\pi_t(a|s)$ is the probability that action $a_t = a$ given that $s_t = s$
- The goal of the agent is to maximize the reward



RL specifies how to change the policy $\pi(a|s)$ given: agent-environment interactions and agent goal

The notion of goal

when there is a finite sequence of actions (episodic tasks), Expected Reward is defined as

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \cdots + r_T$$

For continuous tasks, it makes sense to use a discounting for future rewards:

$$\begin{aligned} R_t &= r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots \\ &= \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \end{aligned}$$

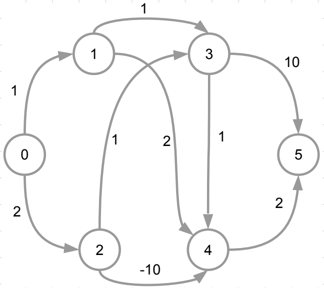
$0 \leq \gamma \leq 1$ is the discount rate

The policy

A policy is a mapping from perceived states to actions to be taken when in those states

1. $0 \rightarrow 1 \rightarrow 3 \rightarrow 5$ = 1 + 1 + 10 = 12
2. $0 \rightarrow 1 \rightarrow 4 \rightarrow 5$ = 1 + 2 + 2 = 5
3. $0 \rightarrow 1 \rightarrow 3 \rightarrow 4 \rightarrow 5$ =1 + 1+ 1+ 2 = 6
4. $0 \rightarrow 2 \rightarrow 4 \rightarrow 5$ = 2 - 10 + 2 = -6
5. $0 \rightarrow 2 \rightarrow 3 \rightarrow 5$ = 2 +1 + 10 = 13
6. $0 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$ = 2 +1 + 1 + 2 = 6

Choose the policy that gives maximum reward

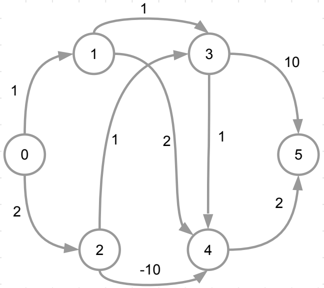


Value function

1. $0 \to 1 \to 3 \to 5 = 12$
2. $0 \to 1 \to 4 \to 5 = 5$
3. $0 \to 1 \to 3 \to 4 \to 5 = 6$
4. $0 \to 2 \to 4 \to 5 = -6$
5. $0 \to 2 \to 3 \to 5 = 13$
6. $0 \to 2 \to 3 \to 4 \to 5 = 6$

$V^1(s_3) = 10$
$V^2(s_3) = 3$

$V^1(s_0) = 12$
$V^2(s_0) = 5$
$V^3(s_0) = -6$
$V^4(s_0) = 13$



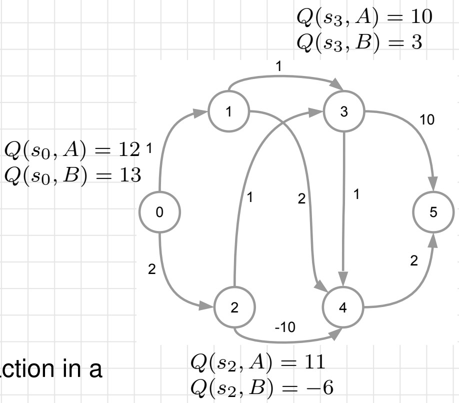
$V^1(s_2) = 11$
$V^2(s_2) = 4$
$V^3(s_2) = -8$

- For a fixed policy define the value as

  how good is it to follow the policy from a given state

Value function

1. $0 \rightarrow 1 \rightarrow 3 \rightarrow 5 = 12$
2. $0 \rightarrow 1 \rightarrow 4 \rightarrow 5 = 5$
3. $0 \rightarrow 1 \rightarrow 3 \rightarrow 4 \rightarrow 5 = 6$
4. $0 \rightarrow 2 \rightarrow 4 \rightarrow 5 = -6$
5. $0 \rightarrow 2 \rightarrow 2 \rightarrow 3 \rightarrow 5 = 13$



- We can also define value of an action in a given state

Value function

State-value function for the policy π

$$V_\pi(s) = \mathbb{E}[R_t | s_t = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s \right]$$

Action-value function for the policy π

$$Q_\pi(s,a) = \mathbb{E}[R_t | s_t = s, a_t = a] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s, a_t = a \right]$$

If we have the value functions (V or Q) the best policy is then

$$\pi(s) = \underset{a}{\operatorname{argmax}}(R(s,a,s') + V_\pi(s'))$$

or

$$\pi(s) = \underset{a}{\operatorname{argmax}}(Q(s,a))$$

Value function can be estimated recursively

State-value function for the policy π

$$\begin{aligned}
V_\pi(s) &= \mathbb{E}[R_t | s_t = s] \\
&= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s\right] \\
&= \mathbb{E}_\pi\left[r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \middle| s_t = s\right] \\
&= \sum_a \pi(a|s) \sum_{s'} p(s'|s,a) \left[r(s,a,s') + \gamma \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \middle| s_{t+1} = s\right]\right] \\
&= \sum_a \pi(a|s) \sum_{s'} p(s'|s,a) \left[r(s,a,s') + \gamma v_\pi(s')\right]
\end{aligned}$$

$p(s'|s,a)$ is the transition probability (or the model of the system) and $r(s,a,s')$ is the expected reward for the transition.

Families of RL appraoches

Model based method : when a model of the environment of transition probs. are known

- Dynamic programming – solve the Bellman equation
- Monte Carlo methods – do not require a model but require experience in the environment i.e. sample sequences of states, actions and rewards

In a more natural setting

- The agent experience is only limited to (s, a, r, s')
- Agent needs to perform actions to generate new experience

Families of RL appraoches

Model based method : when a model of the environment of transition probs. are known

- Dynamic programming – solve the Bellman equation
- Monte Carlo methods – do not require a model but require experience in the environment i.e. sample sequences of states, actions and rewards

In a more natural setting

- The agent experience is only limited to (s, a, r, s')
- Agent needs to perform actions to generate new experience

Model-free approaches to RL

- Temporal difference learning
- SARSA
- Q learning

Families of RL appraoches

Model based method : when a model of the environment of transition probs. are known

- Dynamic programming – solve the Bellman equation
- Monte Carlo methods – do not require a model but require experience in the environment i.e. sample sequences of states, actions and rewards

In a more natural setting

- The agent experience is only limited to (s, a, r, s')
- Agent needs to perform actions to generate new experience

Model-free approaches to RL

- Temporal difference learning
- SARSA
- Q learning

Model-based RL

- Learn the MDP i.e. transition probabilities and associated rewards
- Estimate the transition probability $p(s'|s,a)$ by keeping track of the state action pairs
- Keep track of the reward
- Estimate $V(s)$ and/or $Q(s,a)$ using Bellman equation
- Choose the action that maximizes future value

$$\pi^*(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} (p(s'|a,s)V(s'))$$

Exploration vs Exploitation

Exploration : Take new actions with unknown consequences

- Discover better states than found so far
- estimate better model of the MDP
- Exploration does not guarantee value maximzation
- Something bad may happen

Exploitation : Follow the best strategy so far

- Maximize the reward
- Avoid unexpected bad situations
- Optimal solution may never be found

RL with Exploration

The greedy selection is given as

$$a = \underset{a' \in A(s)}{\operatorname{argmax}} \sum_{s'} (p(s'|s, a')V(s'))$$

Inclusion of exploration means

$$a = \underset{a' \in A(s)}{\operatorname{argmax}} f \left[\sum_{s'} (p(s'|s, a')V(s')), N(s, a') \right]$$

f is the exploration function.
$N(s, a')$ count of action a' taken in state s

Temporal difference learning

In TD algorithms the value function is updated after every time step (*learn a guess from a guess*)

$$V(s_t) \leftarrow V(s_t) + \alpha \Big[r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \Big]$$

The error signal is called the Temporal difference (TD)

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$$

Advantages of TD over other methods

- There is no need of a model as is required in dynamic programming
- The update is online. In Monte-Carlo methods update happens after an episode is over

SARSA: On-policy TD learning

This learning rule uses the quintuple $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$ – hence the name.
Approximate the state-action value function Q

1. Initialize $Q(s,a) = 0 \, \forall s, a$
2. Observe the state s_t
3. Perform action a_t
4. Observe the new state s_{t+1} and reward r_{t+1}
5. Update $Q(s_t, a_t)$

   $$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \, Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right]$$

   α is learning rate

   γ is discount factor for future rewards

Q-value is updated using the Q-value of the next state s' and the action specified by the current policy a^*.

Q learning: Off-policy TD learning

Approximate the state-action value function Q

1. Initialize $Q(s,a) = 0 \, \forall s, a$
2. Observe the state s
3. Perform action a
4. Observe the new state s' and reward r
5. Update $Q(s,a)$

   $$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r + \gamma \max_a Q(s',a) - Q(s_t,a_t)\right]$$

   α is learning rate

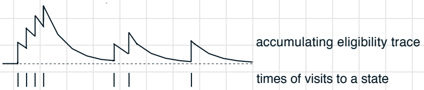
   γ is discount factor for future rewards

   $\max\limits_a Q(s',a)$ is the maximum of Q value in the next state over all actions

TD(λ)

For each state define an eligibility trace

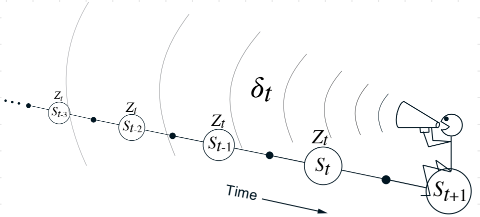
$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) & \text{if} s \neq s_t \\ \gamma\lambda e_{t-1}(s) + 1 & \text{if} s = s_t \end{cases}$$



Each time step the eligibility trace decays by a factor $\gamma\lambda$

The value of each state is updated according to the TD error and eligibility trace

$$\Delta V_t(s) = \alpha\delta_t e_t(s)$$



Actor-critic methods

How to change the policy?

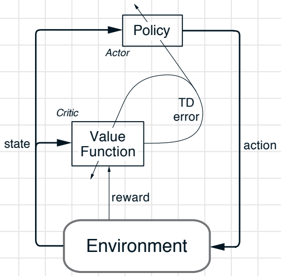
- Policy structure == actor
- Estimated value function == critic

Update the policy according to

$$p(s,a) \leftarrow p(s_t,a_t) + \beta\delta_t$$

Another common policy update is to give credit to the generation of the current action

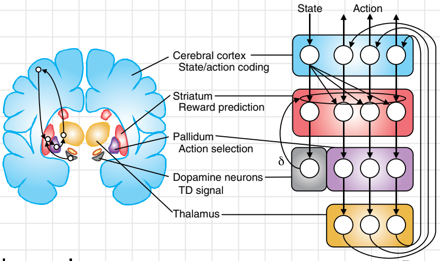
$$p(s,a) \leftarrow p(s_t,a_t) + \beta\delta_t\left[1 - \pi_t(s_t,a_t)\right]$$



RL: main steps

- recognize the present state of the environment by disambiguating sensory inputs
- evaluate the candidate actions in terms of expected future rewards (action values);
- select an action that is most advantageous
- update the action values based on the discrepancy between the predicted and the actual rewards
- update the policy based on the discrepancy between the predicted and the actual rewards

Neural representation of RL



Doya 2008

- Estimate of action value
- Estimate of the reward value r_t
- Action-selection
- Measurement of the temporal derivative δ_t
- Site to add r_t and δ_t
- Locus where error signal can induce plasticity to adapt the prediction

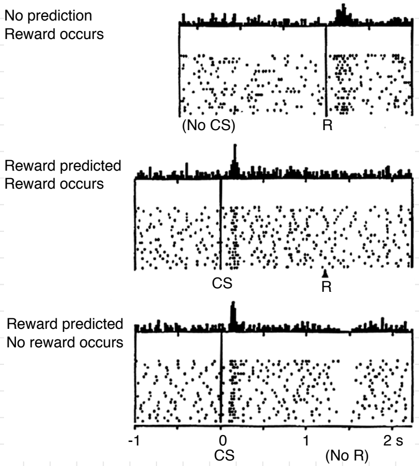
Dopamine as the reward signal

- Addictive drugs (e.g. cocaine) act by enhancing the influence of dopamine on the target neurons
- Dopaminergic neurons are the best target of the electrical self-stimulation – rats choose this stimulation over food/sex
- Dopamine block reduces learning speed
- Dopamine neurons are transiently activated when presented with appetitive stimuli

Dopamine as the reward signal

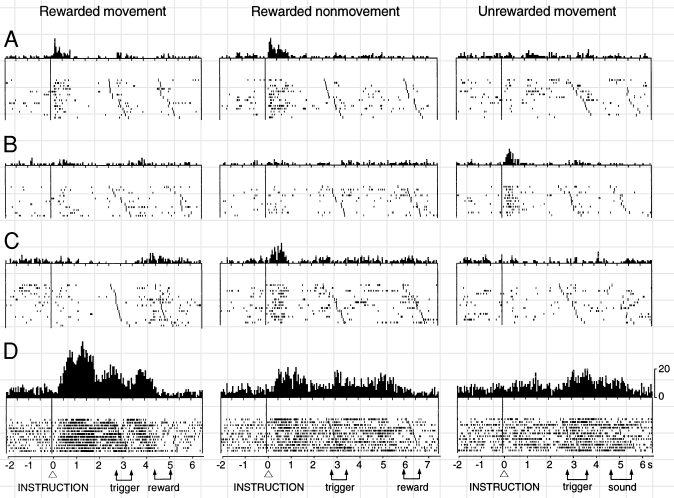
- Dopamine neurons are activated by novel stimuli
- Primary reward no longer elicits the phasic response
- Onset of light results in a phasic response instead
- When the reward does not appear at the expected time, dopamine neuron activity is reduced
- Dopamine encodes the expectation of stimuli or reward (or rather the error signal)
- Positive dopamine signal $\rightarrow$ an appetitive event occurrence is better predicted
- No dopamine signal $\rightarrow$ an appetitive event occurrence is as predicted
- Negative dopamine signal $\rightarrow$ an appetitive event occurrence is worse than predicted

Do dopamine neurons report an error in the prediction of reward?



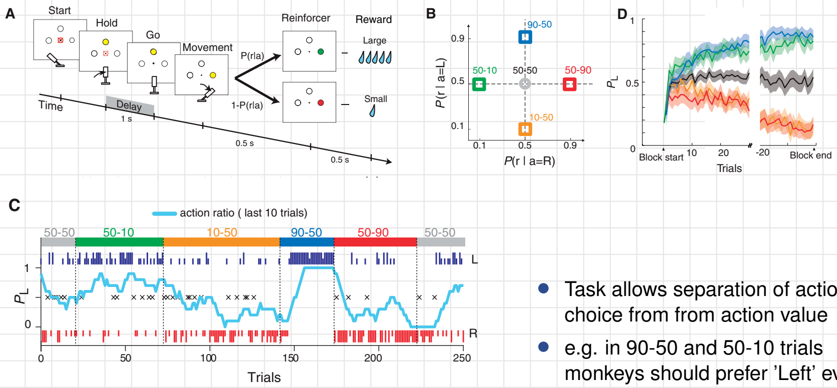
Schultz et al. 1997

Reward related responses in the OFC



Tremblay and Schultz 2000

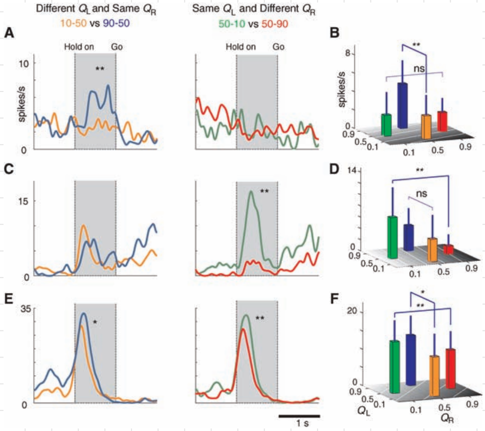
Action specific reward values in the striatum

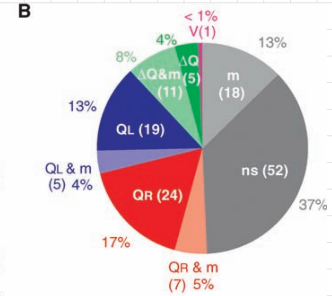


- Task allows separation of action choice from from action value
- e.g. in 90-50 and 50-10 trials monkeys should prefer 'Left' even though Q_L has reduced from 0.9 to 0.5
- e.g. in 90-50 and 10-50 trials monkeys should change the action choice even though Q_R has not changed

Samejima et al. 2008

Action specific reward values in the striatum

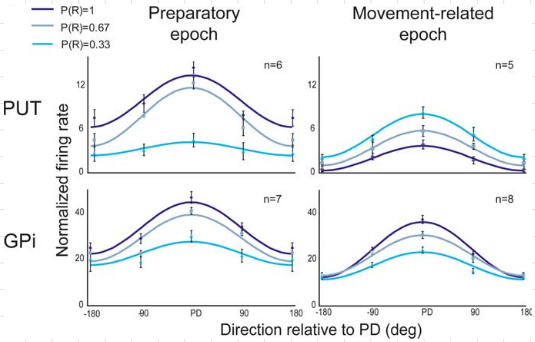




- 142 of 504 neurons from two monkeys showed increased activity
- e.g. in 90-50 and 50-10 trials monkeys should prefer 'Left' even though Q_L has reduced from 0.9 to 0.5
- e.g. in 90-50 and 10-50 trials monkeys should change the action choice even though Q_R has not

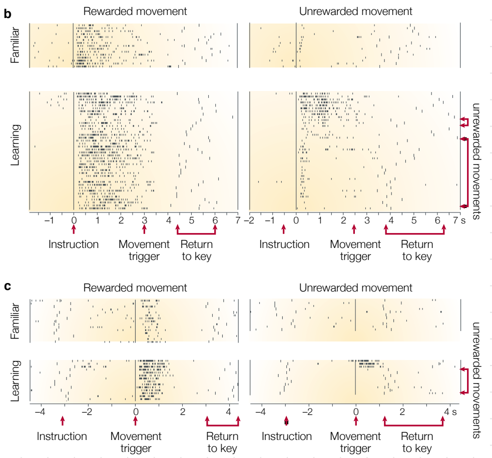
Samejima et al. 2005

Reward expectation modulates stimulus response in the striatum



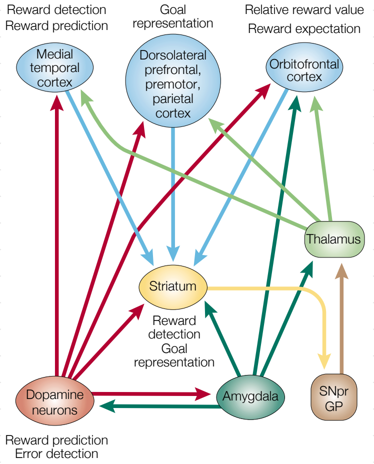
Pasquereau et al. 2007

Reward expectation modulates movement planning in the caudate



Schultz et al. 2000

Multiple reward signals



Schultz et al. 2000

Unresolved issues

- Are model-based and model-free systems separately represented in the brain – how do they interact
- How are the aversive predictions are encoded
- How are the aversive prediction errors are encoded
    - dopamine antagonists suppress aversive prediction error
- Balancing of exploration and exploitation. This requires encoding of uncertainity.
- In tasks where a previously predictive stimulus no longer predict the reward (extinction), animals learn a new predictive association to mask the earlier association
- How do conditioned excitors and conditioned inhibitors interact to generate a single prediction

References

- Sutton & Barto (1998) Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)The MIT Press; 1998.
- Dayan and Niv (2008) Reinforcement learning: The good, the bad and the ugly. *Curr Op. Neurobiol.* 18:185-196
- Daw and Doya (2000) The computational neurobiology of learning and reward. *Curr. Op. Neurobiol.* 16:199?204
- Doya (2008) Modulators of decision making. *Nature Neurosci.* 11:410-416
- Pasquereau et al. (2007) Shaping of motor responses by incentive values through the basal ganglia *J. Neurosci.* 27(5):1176 ?1183
- Samejima et al. (2005) Representation of Action-Specific Reward Values in the Striatum. *Science* 310:1337-1341
- Schultz W (2000) Multiple reward signals in the brain. *Nature Rev. Neurosci.* 1:199-207
- Tremblay & Schultz et al. (2000) Reward-relatedneuronal activity during go-nogo task performance in primate orbitofrontal cortex. *J. Neurophys.* 83: 1864?1876