AMSI
AUSTRALIAN MATHEMATICAL SCIENCES INSTITUTE

National Collaboration in the Mathematical Sciences

Mathematics of Motivated Reinforcement Learning for Humanoid Robot Soccer Head Behaviour

Jake Fountain
University of Newcastle
jake.fountain@uon.edu.au

May 3, 2013

Abstract

The problem of robot localisation involves determining the location of a robot as well as the location of other objects within a world model. Soccer playing robots must localise themselves and a ball on a soccer field in order to play soccer effectively. Humanoid soccer competitions require the competing robots to have humanoid head and vision restrictions. This gives the robot a limited field of view, and presents the problem of *head behaviour*. Which objects will yield the most information when viewed in a given field state? Motivated reinforcement learning techniques were applied to solve the problem of making head behaviour decisions. This paper outlines the mathematics involved with the reinforcement learning techniques used, the motivation procedure implemented and the involved value function approximation methods. The results of the application are detailed in a paper which will be submitted to the 2013 RoboCup Symposium.

1 Robocup and The Head Behaviour Problem

Robocup is an annual international robotics competition with tasks spanning many areas of robotics and artificial intelligence. Among these challenges are the robot soccer leagues (Kitano et al., 1991). The KidSize Humanoid Soccer League is an open platform humanoid soccer league where teams of up to three humanoid robots, with heights between 30 cm and 60 cm, compete on a field of size 6 m by 4 m. The robots must function autonomously while playing the game and all computations must be

performed on the robots' computers. This involves the solution and coordination of many complex computational tasks including computer vision, localisation, locomotion and decision making or behaviour. Vision is responsible for processing image data and producing measurements of objects. This information is then passed to localisation and is incorporated into a world model. There are two types of objects which are important for localisation: *landmarks* and *objects*. Landmarks, such as a goal-post, have a known location in the world model and can be used to localise the robot in the world model. Objects, such as the ball, must be localised in the world model by measuring position relative to a localised robot. The world model maintained by the robot is a probabilistic model, allowing uncertainty to be managed quantitatively for each robot and object. Thus, successful localisation of the robots and field objects relies on the vision system. Furthermore, the vision system relies on the behaviour of the head; limited field of view implies that all objects cannot be measured at once. Thus the head behaviour problem is that of choosing where to look, balancing landmarks and objects, in order to minimise localisation uncertainty. Motivated reinforcement learning techniques were applied to solve this problem.

2 Motivated Reinforcement Learning

Reinforcement learning is a type of machine learning used to solve problems involving a series of decisions based on perceptions, with a metric indicating performance after every decision (Sutton and Barto, 1998). The problem can be formulated as an interaction between a learning entity called the *agent* and its *environment*. The agent is the decision maker and the environment is defined as anything which cannot be directly altered by the agent. The agent measures the state of the environment and must decide an action to take. After each action the agent receives feedback in the form of a reward from the environment. The goal of the agent is to maximise the reward over many actions. This type of problem is called a Markov decision process, and is described fully by: a state space S; a set of actions A and a function $\Lambda : S \to P(A)$ where $\Lambda(s)$ is the subset of actions available in state $s \in S$; a transition function $T : S \times A \times S \to [0, 1]$ describing the probability of state transitions; and a reward function $R : S \times \Lambda(s) \to \mathbb{R}$. Here $P(A)$ is the power set of A, or the set of all subsets of A. It is common for an additional simplification to include $\Lambda(s)$ being finite and discrete for each $s \in S$. To solve this problem the concepts of *policies* and *value functions* are often introduced. A policy $\pi : S \times A \to [0, 1]$ is a function which gives the probability of taking action a from state s as $\pi(s, a)$. Given a policy π, $Q^\pi : S \times A \to \mathbb{R}$ denotes the value function

of the policy. The value of taking action a_0 from state s_0 defined by

$$Q^{\pi}(s_0, a_0) = E[\sum_{i=0}^{\infty} \gamma^i R(s_i, a_i)] \tag{1}$$

where $0 < \gamma < 1$ is called the reward discount factor and s_i is given by taking the action a_{i-1} from state s_{i-1}, chosen according to π, for each $i \geq 2$. Here $E(\chi)$ gives the expectation value of the random variable χ. The *optimal value function* Q^* is then defined by

$$Q^*(s, a) = \max_{\pi} \{Q^{\pi}(s, a)\} \tag{2}$$

Q-learning is a method of learning the optimal value function Q^* and can be performed as an *online* learning method. Actions are chosen from successive states to explore the state-action space and learn the optimal value function. After each action, a stored function $Q : S \times A \to \mathbb{R}$ is updated to approximate Q^* using the following update rule

$$Q(s, a) \leftarrow Q(s, a) + \alpha[R(s, a) - Q(s, a) + \gamma \max_{a' \in \Lambda(s')} Q(s', a')] \tag{3}$$

where s' is given by the transition function T. It has been shown that this method converges to the optimal value function Q^* provided the agent explores all states and each action from each state sufficiently (Sutton and Barto, 1998). Once Q^* is learned, the agent can make informed decisions given each state in an effort to maximise long term reward by taking the action chosen according to

$$a_i = argmax_{a \in \Lambda(s_i)} \{Q^*(s_i, a)\} \tag{4}$$

During learning, actions are chosen either *on-policy* or *off-policy*. Off-policy involves choosing randomly between the possible actions with uniform probability. On-policy involves choosing the action with the highest expected return given by $Q(s, a)$. In practice these methods are combined to balance exploiting known valuable actions and exploring new actions. The ϵ-greedy method involves choosing off-policy actions with probability ϵ and on-policy otherwise. Soft-max involves choosing action a from state s with probability

$$\varphi(s, a) = \frac{e^{Q(s,a)/\sigma}}{\sum_{a \in \Lambda(s)} e^{Q(s,a)/\sigma}} \tag{5}$$

where the parameter $\sigma \in \mathbb{R}$ is called the temperature (Sutton and Barto, 1998). The probability of selecting the highest value action increases with decreasing σ.

Motivated reinforcement learning is based on psychological motivation theory (Merrick and Maher, 2009). An animal removed from environmental stimulation will tend to seek out similar-but-different stimuli (Wundt, 1910). A *motivated* reinforcement learning agent uses internal methods to generate a reward to complement or replace the environmental stimulus. To generate the motivation reward, the *novelty* is calculated by comparing the current state and last action taken to all previous experiences. The novelty $N(s, a)$ is often calculated using methods such as a Habituated Self-Organising Map (Saunders and Gero, 2001), which allows for the novelty to be calculated without storing every past state and action. The method used to calculate novelty for the head behaviour was model based. A model of the expected transition function $T' : S \times A \to S$ was maintained in the form of a Fourier basis linear approximator (see Section 3). After taking action a from state s, the novelty is calculated according to

$$N(s, a) = \|T'(s, a) - s'\| \tag{6}$$

where s' is the state given by the transition function T. Afterward, the expected transition function T' is updated to agree more closely with T. Thus, over time the novelty is reduced when the agent takes an action from a given state repeatedly. This closely reflects the idea of novelty being a measure of how well an agent understands a system. The motivation reward is then calculated according the Wundt function

$$M(s, a) = M_0 + \frac{M_1}{(1 + e^{-\rho_1(N-N_1)})} - \frac{M_2}{(1 + e^{-\rho_2(N-N_2)})} \tag{7}$$

where $N = N(s, a)$ is the novelty of the action a taken from $s \in S$, and ρ_1, ρ_2, N_1, N_2, M_0, M_1 and M_2 are real parameters which define the function's shape (Figure 1).

3 Approximating Continuous Value Functions

To store the functions Q and T' during learning, a *Fourier Basis Linear Approximator* was used. The Fourier basis linear approximator approximates functions $f : \mathbb{R}^m \to \mathbb{R}$ and is given by the cosine part of a truncated Fourier series, updated with a sampled point update rule (Konidaris and Osentoski, 2008). For an approximator $F : \mathbb{R}^m \to \mathbb{R}$ of order $k \in \mathbb{N}$, the value of the approximation at $\mathbf{x} \in \mathbb{R}^m$ is given by

$$F(\mathbf{x}) = \langle \mathbf{w}, \overrightarrow{\phi}(\mathbf{x}) \rangle = \sum_{\mathbf{c} \in X} w_{\mathbf{c}} cos(\frac{\pi}{\tau} \langle \mathbf{c}, \mathbf{x} \rangle) \tag{8}$$

where $X \subseteq (\mathbb{Z}_{k+1})^m$, $\mathbf{w} \in \mathbb{R}^l$ for some $l \in \mathbb{N}$ and $\overrightarrow{\phi} : \mathbb{R}^m \to \mathbb{R}^l$. We say $\phi_{\mathbf{c}}(\mathbf{x}) = cos(\frac{\pi}{\tau} \langle \mathbf{c}, \mathbf{x} \rangle)$ is the basis function corresponding to $\mathbf{c} \in X$ and $w_{\mathbf{c}} \in \mathbb{R}$ is the weight of

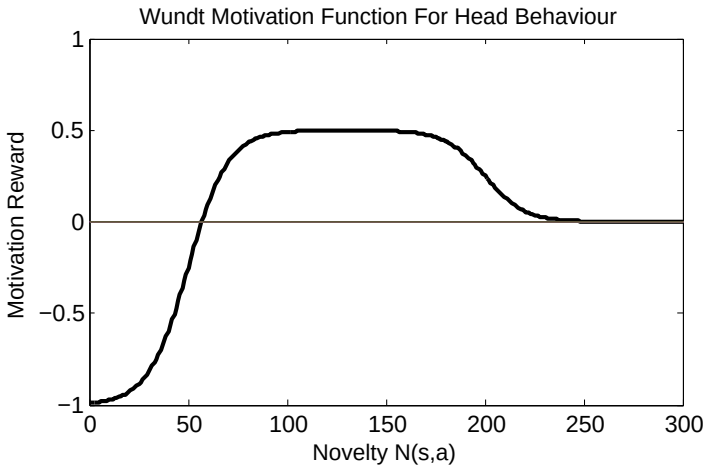

Figure 1: An example of a Wundt function used to calculate the motivation reward from the novelty. The corresponding parameters are $N_1 = 50, N_2 = 200, M_0 = -1, M_1 = 1.5, M_2 = 0.5$ and $\rho_1 = \rho_2 = 0.1$.

the basis function corresponding to $\mathbf{c} \in X$. Here, $\mathbb{Z}_j$ denotes the set of integers modulo j, thus X is a collection of m dimensional integral vectors. $\overrightarrow{\phi}(\mathbf{x})$ is the vector of basis functions; the ordering must simply match that of the weight vector $\mathbf{w}$. $\tau \in \mathbb{R}$ is half of the largest period of the basis functions, which also equals the size of the largest interval over which the approximator will converge, due to the periodicity of the cosine function. A point-wise update rule for the approximator was derived using gradient descent. If $f : \mathbb{R}^m \to \mathbb{R}$ is to be approximated by F, as above, then sampling f at $\mathbf{x} \in \mathbb{R}^m$ allows F to be updated according to

$$\begin{aligned} \Delta\mathbf{w} &= \alpha \frac{\overrightarrow{\phi}(\mathbf{x})}{\|\overrightarrow{\phi}(\mathbf{x})\|^2}[f(\mathbf{x}) - F(\mathbf{x})] \\ \mathbf{w} &\leftarrow \mathbf{w} + \Delta\mathbf{w} \end{aligned} \tag{9}$$

where $\alpha \in [0, 1]$ is called the learning rate. This update rule uses the weight-gradient of the error, $\nabla_{\mathbf{w}}[f(\mathbf{x}) - F(\mathbf{x})] = -\overrightarrow{\phi}(\mathbf{x})$, to update the value of $F(\mathbf{x})$ closer to $f(\mathbf{x})$ along the negative gradient vector in the weight-space. Observe the new value of the

approximator after the update is applied:

$$\begin{aligned} F'(\mathbf{x}) &= \langle \mathbf{w} + \Delta\mathbf{w}, \overrightarrow{\phi}(\mathbf{x}) \rangle \\ &= \langle \mathbf{w}, \overrightarrow{\phi}(\mathbf{x}) \rangle + \langle \Delta\mathbf{w}, \overrightarrow{\phi}(\mathbf{x}) \rangle \\ &= F(\mathbf{x}) + \alpha \frac{\langle \overrightarrow{\phi}(\mathbf{x}), \overrightarrow{\phi}(\mathbf{x}) \rangle}{\| \overrightarrow{\phi}(\mathbf{x}) \|^2} [f(\mathbf{x}) - F(\mathbf{x})] \\ &= (1-\alpha)F(\mathbf{x}) + \alpha f(\mathbf{x}) \end{aligned} \tag{10}$$

Note that $\alpha = 1$ gives $F'(\mathbf{x}) = f(\mathbf{x})$, as required. However, in practice multiple updates with low α gives better convergence due to higher resolution sampling of the gradient vector. To approximate a function $g : \mathbb{R}^m \to \mathbb{R}^n$, n Fourier basis approximators are used, one for each component of the output. This technique is essentially a method for learning a truncated Fourier series expansion of a function.

4 Experimental Outline

A series of agents were trained to solve a reinforcement learning problem based on the soccer field state and actions given by looking at certain objects on the field. Environmental rewards were generated based on the localisation uncertainty of the world model. An agent which combined the environmental reward with a motivation reward performed the best at localising on the soccer field. A detailed report on the results is planned for submission to the RoboCup Symposium 2013.

Acknowledgments. This project was supported by the Australian Mathematical Sciences Institute (AMSI) summer research scholarship program.

Thanks to the NUbots, the University of Newcastle's RoboCup team, for their support and the use of their hardware.

References

Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, Eiichi Osawa, and Hitoshi Matsubara. Robocup: A challenge problem for ai. *AI Magazine*, 18(1), 1991.

George Konidaris and Sarah Osentoski. Value function approximation in reinforcement learning using the fourier basis. *Computer Science Department Faculty Publication Series, University of Massachusetts*, 2008.

Kathryn E. Merrick and Mary Lou Maher. *Motivated Reinforcement Learning: Curious Characters for Multuser Games*. Springer, Dordrecht, 2009.

Rob Saunders and John S. Gero. Designing for interest and novelty - motivating design agents. In Bauke de Vries, Jos van Leeuwen, and Henri Achten, editors, *Proceedings of the ninth international conference on Computer aided architectural design futures*, pages 725–738. Kluwer Academic Publishers, 2001.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

Wilhelm Wundt. *Principles of Physiology and Psychology*. Macmillan, New York, 1910.