Fluent Values

Here's a formulation of fluent calculus that unifies perception with robot actions.

Learning a knowledge representation

We assume that at any given point in time, the world is in a particular *state*. Reasoning is done on *fluents*, which are functions on states. An agent is driven by its *values* to cause fluent changes.

States

An artificial agent a decodes visual v and linguistic l input to infer the most likely state s of the world. The agent receives the sensory inputs in parallel at each time-step t. Knowledge is captured in a graph G that contains spatial G_S, temporal G_T, and causal G_C concepts. The following is the likelihood of G given $\mathcal{D} = \langle v, l, g' \rangle$ (g' is the previous knowledge of the world).

$$P(G|\mathcal{D}) = P(G_S, G_T, G_C|\mathcal{D}) = P(G_S|\mathcal{D})\ P(G_T|G_S, \mathcal{D})\ P(G_C|G_S, G_T, \mathcal{D}) \tag{1}$$

Knowledge acquisition (for agent a) can be formulated by maximum likelihood over all STC-AOGs, G, to sample a parse-graph pg. Figure 1 diagrams the knowledge acquisition scheme.

$$s_a^{(t)} = pg_a^{(t)} \sim g_a^{(t)} = \operatorname*{argmax}_g P_G(\ G = g \mid v_a^{(t)}, l_a^{(t)}, g_a^{(t-1)}) \tag{2}$$

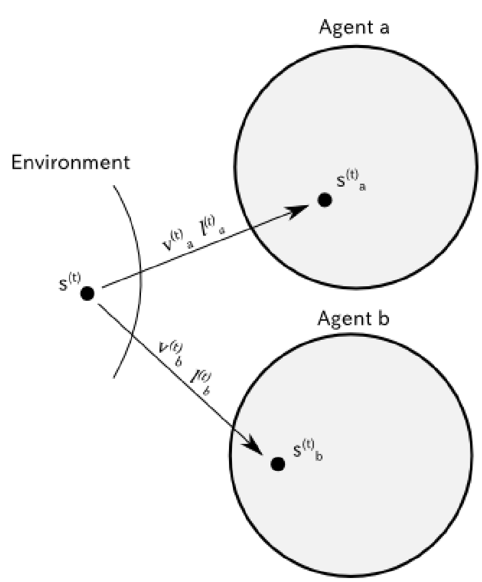

Figure 1: Knowledge is acquired through a noisy medium of vision and language.

Fluents

A fluent f is a function on a state $s^{(t)}$. We assume there is a large number N of fluents, and index f_i for $i \in \{1, ..., N\}$.

A fluent-vector $F = (f_1, ..., f_N)$ on a state $s^{(t)}$ is a vector of all fluents f_i stacked together.

An action is defined as a change in a fluent-vector $\triangle F$.

Values

The "value of a state" $V(s)$ is defined by a potential function U on the fluent-vector.

$$V(s^{(t)}) = U(F(s^{(t)})) \in \mathbb{R} \tag{3}$$

We assume U is linear, so each fluent f_i has a corresponding weight $\omega_i \in \mathbb{R}$. The value of a state is computed by the learned ω_i weights.

$$V(s^{(t)}) = \omega^\intercal F(s^{(t)}) = \sum_{i=1}^{N} \omega_i f_i(s^{(t)}) \tag{4}$$

An agent learns the value of states by witnessing fluent-changes $F(s^{(t)}) \rightarrow F_2(s^{(t+1)})$. The learning problem boils down to minimizing $||\omega||_1$ that satisfy the following constraints:

- If a relevant action caused $||\frac{f_i(s^{(t)})}{dt}|| > ||\frac{f_j(s^{(t)})}{dt}||$ then $\omega_i > \omega_j$.
  "A fluent that changes due to a relevant action should have a higher weight."
- If an irrelevant action caused $||\frac{f_i(s^{(t)})}{dt}|| > ||\frac{f_j(s^{(t)})}{dt}||$ then $\omega_i < \omega_j$.
  "A fluent that changes due to a irrelevant action should have a lower weight."

Inferring actions from learned model

The inference step performs an action $\triangle F$ given the current state s. The action $\triangle F$ is independent of the actuators of an agent. The goal of action selection is to minimize the amount of work required to achieve $\triangle F$.

Let Ψ be a random variable representing a possible action the robot can take. Inferring which action to perform is also a function of the previous $m \in \mathbb{N}$ actions and the current state s. All together, it's formulated as a maximum likelihood estimate:

$$\psi_a^{(t+1)}(\triangle F) = \underset{\psi}{\operatorname{argmax}}\, P_\Psi(\ \Psi = \psi \mid s_a^{(t)}, \triangle F, \psi_a^{(t-m)}, ..., \psi_a^{(t)}) \tag{5}$$

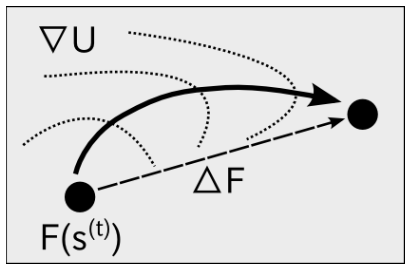

Figure 2: To achieve a fluent change, the agent follows the gradient of values.