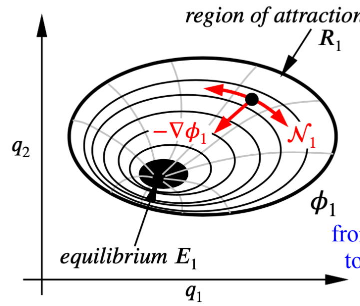
The Control Basis Primer:

Landscape of Attractors
Markov Decision Processes
States, Actions $\Phi \times \Sigma \times \mathcal{T}$, SEARCH & TRACK
Markovian Supervisory Structure
Q-learning (value iteration)
Risk Sensitive Control (policy iteration)

Closed-Loop Control Abstraction



status $\in \{0,\ 1,\ 2\}$

actions: $\phi|_{\tau}^{\sigma}$

- ϕ : a potential function
- σ : sensor
- τ : motor units

from now on, our focus is on how to give setpoints to motor units

Inventory

motor units:
- void PCcontroller_eyes(roger, time)
- void PDController_arms(roger, time)
- void PDBase_translate(roger, time)
- void PDBase_rotate(roger, time)
- void PDController_base(roger, time)

tools:
- void fwd_arm_kinematics( )
- int inv_arm_kinematics( )
- int avg_red_pixel( )
- void stereo_observation( )

skills:
- int TrackBall( ) /* aka Track( ), eyes, eyes+base */
- int SearchTrackBall( ) /* aka SearchTrack( ), eyes+base */
- int Chase( )
- int Touch( )
- int ChaseTouch( )

Basic Supervisory Structure: SearchTrackBall( )

```
/* file:     project4.c                                                  */
/* global variables, scope is a “project” i.e. a single task supervisor  */
#define SEARCHBALL 0
#define TRACKBALL   1

for each action a ∈ A {       /* for both SearchBall( ) & TrackBall( )   */
   for each dof { r_DiffSetpt[action][dof] = 0.0; } }

void project4_control(roger, time) {
  state = 0;
  /* execute all  a ∈ A return status & new recommended setpoints */
  SearchBall( );          /* assigns r_DiffSetpt[0][NDOF], no return status d ∈ {0, 1, 2}    */
  /* Roger observes states directly! */
  d = TrackBall( );       /* assigns r_DiffSetpt[1][NDOF], plus returns status d ∈ {0, 1, 2} */

  /* make a control decision */
  switch(TrackBall( )) {
     case(0): submit r_DiffSetpt[SEARCHBALL] to motor units; break
     case(1): submit r_DiffSetpt[TRACKBALL] to motor units; break
     case(2): submit r_DiffSetpt[TRACKBALL] to motor units; break
    }
}
return d;                      /* SearchTrackBall( ) returns TrackBall( ) status d ∈ {0, 1, 2} */
```


Basic Supervisory Structure: In Practice

concurrency:
in some cases, you can submit more than one recommendation
for example, Chase( ) uses the mobile base
Touch( ) uses an arm
disjoint degrees of freedom can receive setpoints simultaneously
(they treat each other as perturbations)

new_setpoint =
current_setpt + r_DiffSetpt[CHASE] + r_DiffSetpt[TOUCH]

there is another way to compose concurrent controllers in the Chapter

in most cases, Track( )'s are subsumed by SearchTrack( )'s

The Control Basis

The newborn uses a maturational schedule for engaging sensory and motor resources to throttle the computational complexity of learning to interact with unstructured environments.

In place of a relatively small set of special purpose developmental reflexes, an exhaustive array of closed-loop control relations is proposed that tile a high dimensional state space with multiple lower-dimensional attractors.

to support programming and machine learning, the actions in the architecture must be *parameterized by resources*, *asymptotically stable* and *composable*

Control Basis: a parametric Landscape of Attractors

T: TRACK

$$a = \phi|_{\tau}^{\sigma}$$

closed-loop feature TRACK-er with (σ, τ) resources

σ : sensor resources that establish concrete observable "facts" about the environment

τ : motor resources actuate kinematic chains controlling sensor viewpoints

state : $\gamma(a)$ membership functions in the phase portrait $(\phi, \dot{\phi})$ over empirical models

- a framework for stochastic exploration over resource combinations (σ, τ) that establish a control context in response to environmental stimuli

S: SEARCH

$$a = \phi|_{\tau}^{\tilde{\sigma}}$$

- the *orient* counterpart of TRACK actions
- establish a probabilistic basis for environmental structure

$\tilde{\sigma}$ sampled from $Pr(\boldsymbol{u}_\tau | \gamma(\phi|_{\tau}^{\sigma}))$

Control Basis: SEARCH primitive

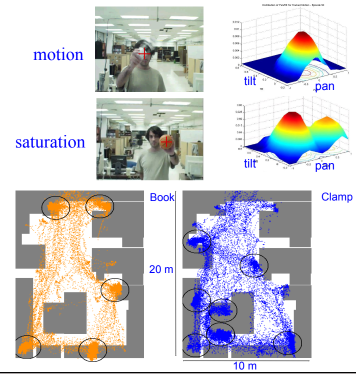
S: SEARCH

$$a = \phi|_{\tau}^{\tilde{\sigma}}$$

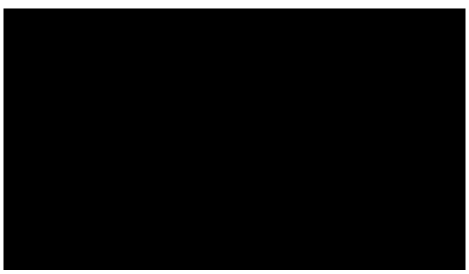
$$\tilde{\sigma} = Pr(\boldsymbol{u}_\tau | \gamma(\phi|_{\tau}^{\sigma}) \Rightarrow conv)$$

(TRACK)

- the *orient* counterpart of TRACK actions

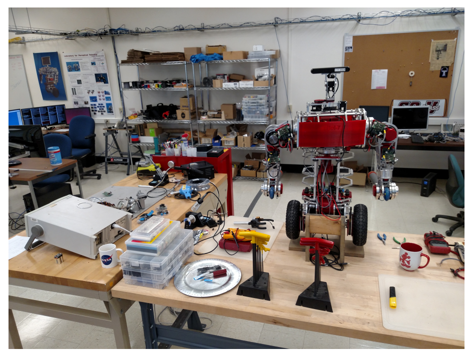


EXAMPLE: Coordinated Human-Robot Search



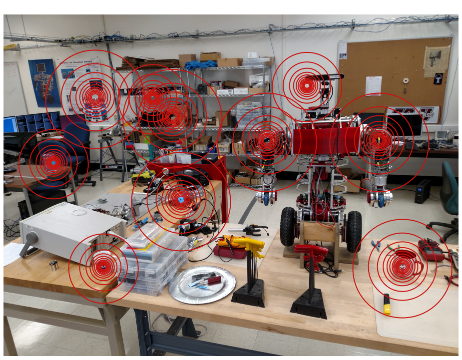
on average, HR team performed 40% better than human alone

Landscapes of Attractors



Where's my coffee cup?

Landscapes of Attractors



Landscapes of Attractors – visual features

serve as goals for oculomotor controls

parse the scene using geometric structure in constellations of visual features

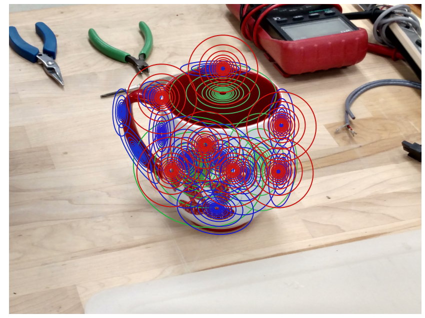
Landscapes of Attractors - tactile

serve as goals for arm/hand controls

parse the scene using geometric structure in constellations of tactile features

UMassAmherst

Landscapes of Attractors – multimodal



a multimodal goal set defining my coffee cup

parse the scene using geometric structure in constellations of multmodal features

UMassAmherst

Markov Decision Processes

$M = < S, A, T, R >$

S : set of system states

A : set of available actions

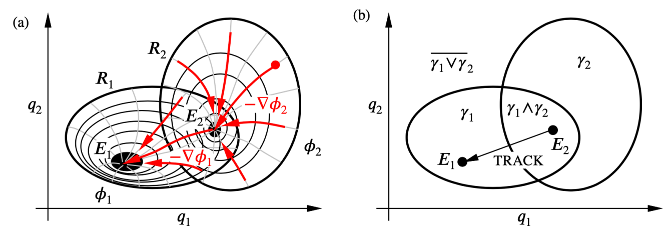
$T : S \times A \times S \mapsto [0,1]$ probability that (s_k, a_k) transitions to state s_{k+1}

$R : S \times A \mapsto \mathbb{R}$ real-valued reward for (s_k, a_k)

definition of a memory-less stochastic process
probability of future states depends solely on the current state

UMassAmherst

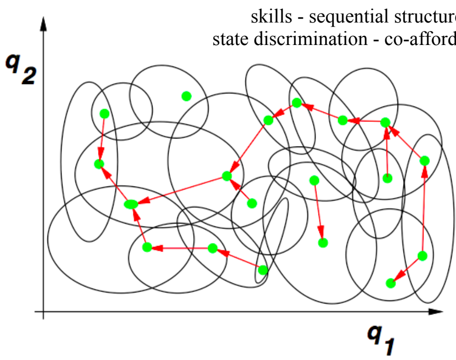
State Transitions - Probabilistic Models of the Environment



UMassAmherst

Tiling the State Space with Skills

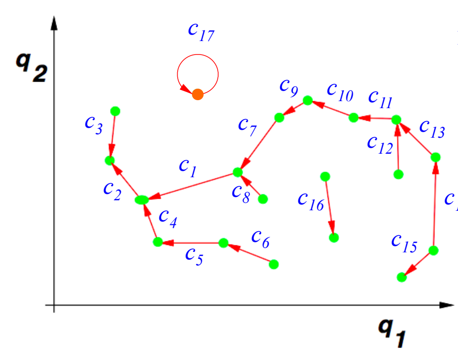
skills - sequential structure
state discrimination - co-affordances



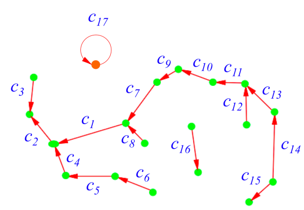
UMassAmherst

Tiling the State Space with Skills

LMT 1984
Burridge 1999
Tedrake 2009



Tiling the State Space with Skills



skills - sequential structure
state: [g_1 … g_n]

"objects": transition dynamics
reference to places (q_1,q_2) not required

some of these are referenced to stimuli in the environment that move in (semi)rigid groups … "objects," "rooms," etc

joint distributions/graph homomorphisms convey important context information

Example: Reading a Bar Code

sequential, multi-objective control

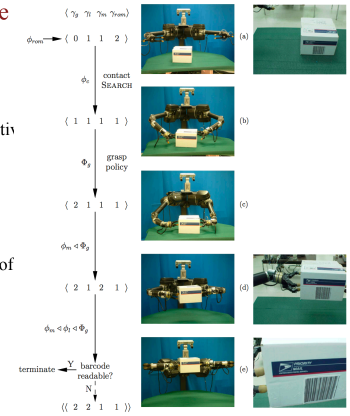
φ_{rom}: the Postural range of motion objectiv

φ_c : Search for contact signals;

Φ_g: a grasp control policy

φ_m: a Postural bimanual manipulability of the arms; and

φ_l: a Postural localizability controller



Potential Functions

- The value of a scalar potential at the location of a particle in a field represents the energy that will be liberated if the particle is released from this configuration.

  e.g. the gravitational potential of a particle of mass *m* near the Earth is the work required to move particle from the surface of the Earth to altitude *h*.

$$\phi_g = \int \mathbf{F} d\mathbf{s} = \int_0^h (mg) dz = mgh$$

The gradient of the potential field defines a force acting on the particle that returns the system to its equilibrium state.

$$\mathbf{F}_g = -\nabla mgh = -(mg)\,\hat{\mathbf{z}}$$

Potential Functions – Spring-Mass-Damper

- For the SMD, the potential function is the energy stored in the spring

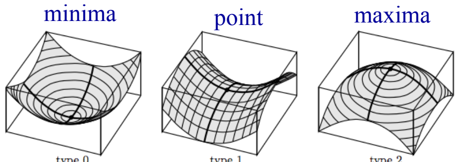
$$\phi_K = \int \mathbf{F} d\mathbf{s} = \int_0^x (Kx) dx = \frac{1}{2} K x^2$$

which is released when the spring is allowed to assume its original shape

$$\mathbf{F}_K = -\nabla \phi = -(Kx)\,\hat{\mathbf{x}}$$ Hooke's law

Equilibrium Point Theory - Differential Geometry

minima saddle point maxima



stable *fixed* points unstable critical points

Critical points – places where the gradient vanishes

$$\nabla \phi = \left[\frac{\partial \phi}{\partial q_0} \;\; \frac{\partial \phi}{\partial q_1} \;\; \cdots \;\; \frac{\partial \phi}{\partial q_n} \right] = \mathbf{0}.$$

Potential Functions and Local Minima

Curvature in the Neighborhood of a Critical Point

$$\frac{\partial^2 \phi}{\partial \mathbf{q}^2} = \begin{bmatrix} \frac{\partial^2 \phi}{\partial q_1^2} & \frac{\partial^2 \phi}{\partial q_1 \partial q_2} & \cdots & \frac{\partial^2 \phi}{\partial q_1 \partial q_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \phi}{\partial q_n \partial q_1} & \frac{\partial^2 \phi}{\partial q_n \partial q_2} & \cdots & \frac{\partial^2 \phi}{\partial q_n^2} \end{bmatrix}$$

a critical point is said to be **degenerate** if it also has zero curvature

excluding degenerate critical points, gradient descent will converge to type 0 critical points exclusively

Potential Functions and Local Minima

$$\frac{\partial^2 \phi}{\partial \mathbf{q}^2} = \begin{bmatrix} \frac{\partial^2 \phi}{\partial q_1^2} & \frac{\partial^2 \phi}{\partial q_1 \partial q_2} & \cdots & \frac{\partial^2 \phi}{\partial q_1 \partial q_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \phi}{\partial q_n \partial q_1} & \frac{\partial^2 \phi}{\partial q_n \partial q_2} & \cdots & \frac{\partial^2 \phi}{\partial q_n^2} \end{bmatrix}$$

convex – if the Hessian of ϕ is positive semi-definite over domain **q**, it has ≤ 1 stable fixed points on the interior of **q**

harmonic –
if the trace of the Hessian $\nabla^2 \phi = \frac{\partial^2 \phi}{\partial q_1^2} + \frac{\partial^2 \phi}{\partial q_2^2} + \cdots + \frac{\partial^2 \phi}{\partial q_n^2} = 0.$
then ϕ has no local minima

Harmonic Functions

$$\nabla^2 \phi = \frac{\partial^2 \phi}{\partial q_1^2} + \frac{\partial^2 \phi}{\partial q_2^2} + \cdots + \frac{\partial^2 \phi}{\partial q_n^2} = 0.$$

soap films, laminar fluid flow, steady state temperature in thermally conductive media, voltage distribution in electrically conductive media,

- exclude local minima (and maxima)
- only type 1 critical points (saddle points) (sets of measure zero)
- gradient flow produces non-intersecting *streamlines*
- *hitting probability* of a random walk --- use in path planning

Navigation Functions

analyticity - infinitely differentiable (C^∞ continuous) such that its *Taylor series* about q_0 converges to ϕ (q) for q in the neighborhood of q_0.

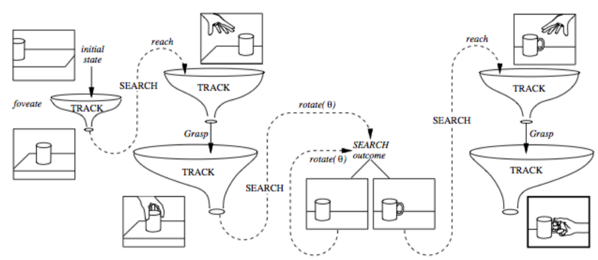
polar - gradients (streamlines) terminate at a unique minimum.
functions that contain type 1 minima exclusively are polar

Morse – functions whose isolevel curves are single points, closed curves, or closed curves that join at critical points …
Morse functions cannot include degenerate critical points

admissibility - Potential fields for robot control require *bounded torque* at obstacle boundaries (and everywhere else in the interior subset of configuration space as well).

Funnel-ing the World Using TRACK/SEARCH Actions

Grasp the cup's handle with the right hand.



Models - learn the parameters of SEARCH actions that transition between multiple TRACK affordances

Control Basis: TRACK primitive

objectives x *sensors* x *effectors*

T: TRACK

$a = \phi|_\tau^\sigma$

action: closed-loop feature (σ) tracker where sensor viewpoint is controlled with kinematic chain τ

ϕ satisfies the important properties of a navigation function

... - visual - auditory – contact force - ...
any feature of any signal whose source is in the 3D world

The Track( ) Control Jacobian

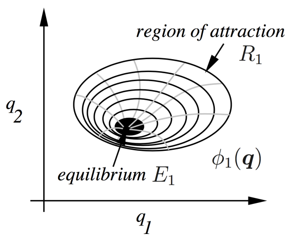
$$\mathbf{J}_c = \frac{d\phi(\sigma)}{d\boldsymbol{u}_\tau} = \left[\frac{\partial \phi(\sigma)}{\partial u_1} \; \frac{\partial \phi(\sigma)}{\partial u_2} \; \cdots \; \frac{\partial \phi(\sigma)}{\partial u_n} \right]_{1 \times n}$$

$$\begin{aligned} \Delta \boldsymbol{u}_\tau &= \kappa \, \mathbf{J}_c^\# \, (\phi_{ref} - \phi(\sigma)), \quad \text{and if } \theta_{ref} = 0, \\ &= -\kappa \, \mathbf{J}_c^\# \, \phi(\sigma), \end{aligned}$$

$$\gamma(\phi, \dot{\phi}) = \begin{cases} \text{NO\_REFERENCE}: & \sigma \text{ undetected} \\ \text{TRANSIENT}: & |\mathbf{J}_c| > \epsilon \quad (\text{or } \nabla\phi > 0) \\ \text{CONVERGED}: & |\mathbf{J}_c| \leq \epsilon. \quad (\text{or } \nabla\phi = 0) \end{cases}$$

UMassAmherst

Summary: Control Basis TRACK Actions



$\phi\,|^{\sigma}_{\tau}$ ← ENV →

scalar

$$J = \frac{\partial \phi(\sigma)}{\partial u_\tau} \in R^{1\times n}$$

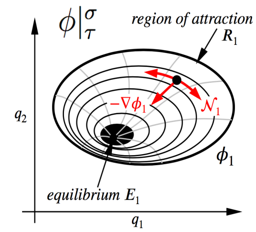
vector of changes in setpoints

m < n
fewer rows than columns
redundant (underconstrained)

UMassAmherst

Multi-Objective Control

$$\begin{aligned}\Delta \mathbf{u}_\tau &= \mathbf{J}_1^{\#}\Delta\phi_1(\sigma_1) + \mathcal{N}_1\left(\mathbf{J}_2^{\#}\Delta\phi_2(\sigma_2)\right),\\ &= \mathbf{J}_1^{\#}\Delta\phi_1(\sigma_1) + [\mathbf{I} - \mathbf{J}_1^{\#}\mathbf{J}_1]\left(\mathbf{J}_2^{\#}\Delta\phi_2(\sigma_2)\right),\end{aligned}$$

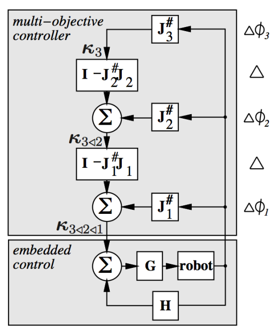


where, $\mathcal{N}_1 = [\mathbf{I} - \mathbf{J}_1^{\#}\mathbf{J}_1]$

the annihilator of J
Appendix A.9

UMassAmherst

Multi-Objective Control



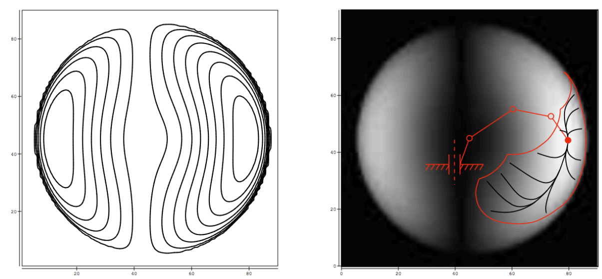
$$\boldsymbol{\kappa}_3 = \mathbf{J}_3^{\#}\Delta\phi_3$$

$$\boldsymbol{\kappa}_{3\triangleleft 2} = \mathbf{J}_2^{\#}\Delta\phi_2 + (\mathbf{I}_3 - \mathbf{J}_2^{\#}\mathbf{J}_2)\boldsymbol{\kappa}_3$$

$$\boldsymbol{\kappa}_{3\triangleleft 2\triangleleft 1} = \mathbf{J}_1^{\#}\Delta\phi_1 + (\mathbf{I}_3 - \mathbf{J}_1^{\#}\mathbf{J}_1)\boldsymbol{\kappa}_{3\triangleleft 2}$$

UMassAmherst

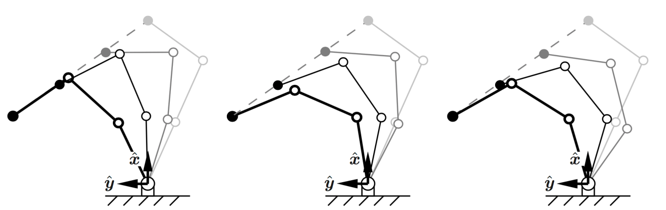
Control Basis: POSTURAL primitive



$$\phi(x,y) = \max_{\boldsymbol{\theta}} \sqrt{det(\mathbf{J}\mathbf{J}^T)}$$

UMassAmherst

Combining SEARCH and TRACK



(a) ϕ_{xy} (b) $\phi_{rom} \triangleleft \phi_{xy}$ (c) $\phi_m \triangleleft \phi_{xy}$

tendon routing in
the human finger

UMassAmherst

ControlBasis-II

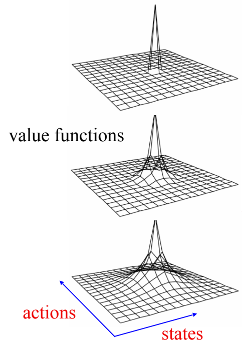
automating sequential control composition

- reinforcement learning
  - *example: hierarchical walking*
- Dynamic Motion Primitives (DMPs)
- policy search

Conditioned Response

Pavlov, I. P. (1927), "Conditioned Reflexes: An Investigation of the Physiological Activity of the Cerebral Cortex," Translated and Edited by G. V. Anrep. London: Oxford University Press.

A Computational Model for Conditioned Response



value functions - an generalization of the potential field

Reinforcement Learning - *value* iteration

- "diffusion" processes
- *curse of dimensionality* diminished by exploiting neurological structure

Markov Decision Processes

describe a memory-less stochastic process

the conditional probability distribution of future states of the process depends only on the current/present state---not how the process got to this state.

The Bellman Equation

Define a policy, $\pi(s, a)$, to be a function that returns the probability of selecting action $a \in A$ from state $s \in S$

the value of state s under policy π, denoted $V_\pi(s)$, is the expected sum of discounted future rewards when policy π is executed from state s,

$$V^\pi(s) = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right\}$$

$0.0 < \gamma \leq 1.0$ represents a discounting factor per decision, and scalar r_t is the reward received at time t.

The Bellman Equation

$$\begin{aligned} V^\pi(s) &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right\} \\ &= E_\pi \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s \right\} \\ &= \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_{t+1} = s' \right\} \right] \\ &= \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^\pi(s') \right] \end{aligned}$$

The Bellman Equation

$$V^\pi(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^\pi(s') \right]$$

$$V^*(s) = \max_\pi V^\pi(s)$$

$$V^*(s) = \max_a \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^*(s') \right]$$

Value Iteration

Dynamic Programming (DP) algorithms compute optimal policies from complete knowledge of the underlying MDP

Reinforcement Learning (RL) algorithms are an important subset of DP algorithms that do not require prior knowledge of transition probabilities in the MDP.

$$V^*(s) = \max_a \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^*(s')\right]$$

provides the basis for a numerical iteration that incorporates the Bellman consistency constraints to estimate $V^\pi(s)$.

$$V_{k+1}(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V_k(s')\right]$$

a recursive numerical technique that converges to V^π as $k \to \infty$

Q-learning

$$V_{k+1}(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V_k(s')\right]$$

Typically, DP employs a full backup--- a comprehensive sweep through the entire state-action space using numerical relaxation techniques (Appendix C).

RL techniques generally estimate $V^\pi(s)$ using sampled backups at the expense of optimality guarantees.

Attractive in robotics because it focuses exploration on portions of the state/action space most relevant to the reward/task

greedy ascent of the converged value function is an optimal policy for accumulating reward.

Q-learning

Quality function – the value function written the state/action form

$$Q^\pi(s,a) = \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V_k(s')\right]$$

Policy improvement:

$$Q(s,a)_{k+1} \leftarrow Q(s,a)_k + \alpha \left[r(s') + \gamma \max_a Q(s',a) - Q(s,a)\right]$$

The policy improvement theorem guarantees that a procedure like this will lead monotonically toward optimal policies.

Q-learning

a natural paradigm for composing skills using the control basis actions because it can construct policies using sequences of actions by exploring control interactions *in situ*.

policy $\quad \pi(s) = \operatorname{argmax}_{a_i} Q(s, a_i)$

maps states to optimal actions by greedy ascent of the value function.

Example: Learning to Walk (ca. 1996)

THING Quadruped
four coordinated robots
2^{13} states × 1885 actions

Resource Model

sensor resources -
- configuration of legs {0123}
- configuration of body (x,y,θ)

effector resources -
- configuration of legs {0123}
- configuration of body (x,y,θ)

control types -
- moment control
- kinematic conditioning

Example: Walking Gaits

13 controllers

moment control $\quad \Phi_{1a}^{abc} \quad abc \in \{0123\}$

kinematic conditioning $\quad \Phi_{2\varphi}^{0123}$

total of 1885 concurrent control options

discrete events:

$p_0 \leftarrow \Phi_*^{012}$

$p_1 \leftarrow \Phi_*^{023}$

$p_2 \leftarrow \Phi_*^{123}$

$p_3 \leftarrow \Phi_*^{013}$

$p_4 \leftarrow \Phi_\varphi^{0123}$

Example: Behavioral Logic for Development

propositions that constrain patterns of discrete events in the dynamical system

Platform stability constraints

- at least 1 of 4 stable tripod stances to be true at all times

$$p_0 \vee p_1 \vee p_2 \vee p_3$$

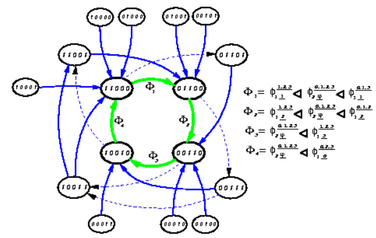
- kinematic constraints

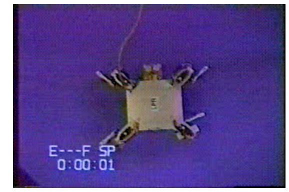
$$\neg(p_0 \wedge p_1) \wedge \neg(p_2 \wedge p_3)$$

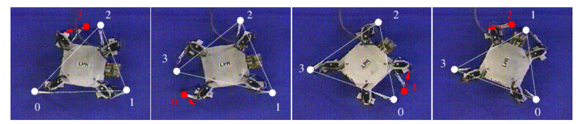
reduced model:

- 32 states × 157 actions
- reduced by 99.94 %

Example: ROTATE schema

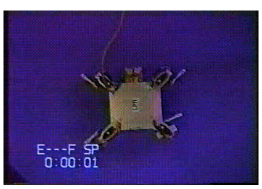




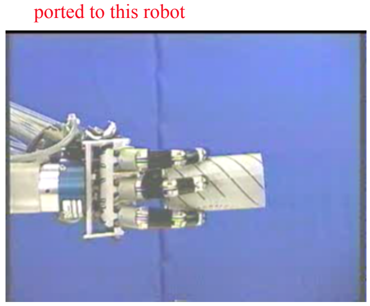


Transfer

"written" by this robot



ported to this robot



Implications of Developmental Hierarchy

native control basis primitives

$\mathcal{A}_0 = \{\phi_1, \phi_2, \phi_3\}$

stage 1 ↓ $\mathcal{C}_1 = \{\phi_1, \phi_2\}$

$\mathcal{A}_1 = \{\phi_1, \phi_2, \phi_3, \underline{\text{ROTATE}}\}$

stage 2 ↓ $\mathcal{C}_2 = \{\phi_1, \phi_2, \text{ROTATE}\}$

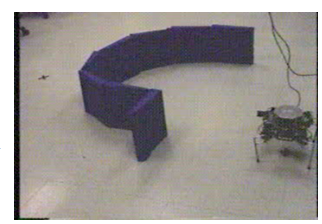
$\mathcal{A}_2 = \{\phi_1, \phi_2, \phi_3, \text{ROTATE}, \underline{\text{STEP}}\}$

stage 3 ↓ $\mathcal{C}_3 = \{\phi_1, \phi_2, \text{ROTATE}, \text{STEP}\}$

$\mathcal{A}_3 = \{\phi_1, \phi_2, \phi_3, \text{ROTATE}, \text{STEP}, \underline{\text{WALK}}\}$

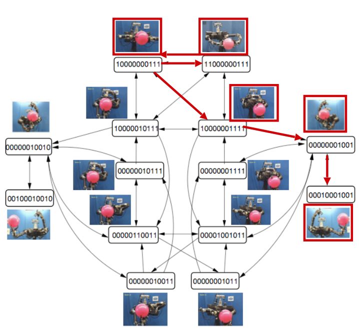
stage 4 ↓ $\mathcal{C}_4 = \{\phi_3, \text{WALK}\}$

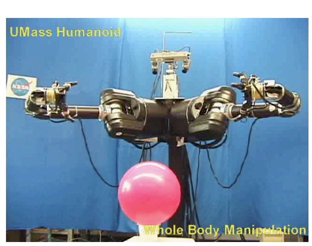
$\mathcal{A}_4 = \{\phi_1, \phi_2, \phi_3, \text{ROTATE}, \text{STEP}, \text{WALK}, \underline{\text{NAVIGATE}}\}$



"Objects" - Fully-Observable Case (Aspect Transition Graph)

Rob Platt



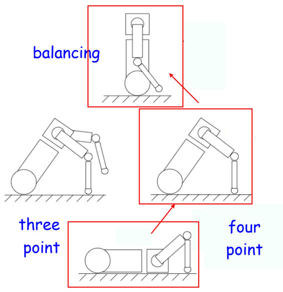


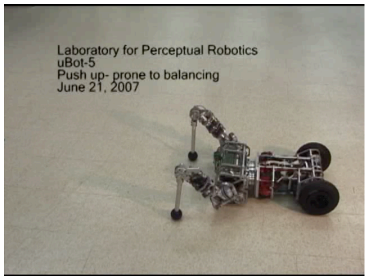
...at least one stable grasp must exist at all times...$(\gamma(\Phi_{g1}) \vee \gamma(\Phi_{g2}) \vee \gamma(\Phi_{g3}))$

Assembly

feature-level milestones for planning and learning

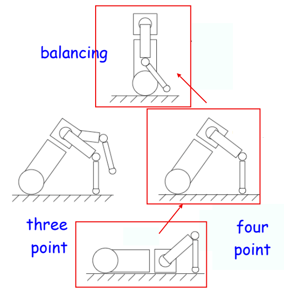
Hierarchical Commonsense Control Knowledge





γ(prone) V γ(4-point) V γ(balance)

Hierarchical Commonsense Control Knowledge



γ(prone) V γ(4-point) V γ(balance)

Affordance Modeling - Three Objects

exploration habituates when model stops changing

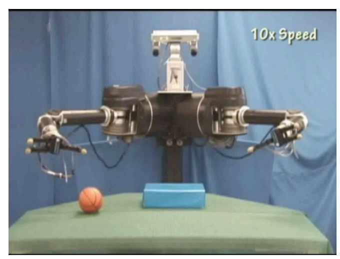
visual hue tracker

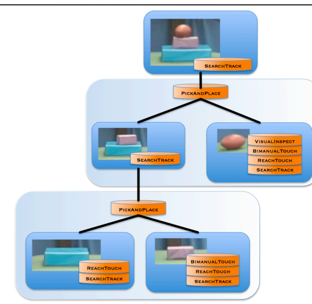
grasp

pick-and-place

Stephen Hart

Modeling Simple Assemblies





stable *multi-body* relations

Human Tracking

- disambiguate human structure against cluttered backgrounds
- references (hands/face) for control behavior and modeling

Expressive Communicative Actions

learning about kinodynamic and intentional agents

unreachable objects can be reachable in the presence of a human being, the dynamics of the world change

Receptive Pointing Skills