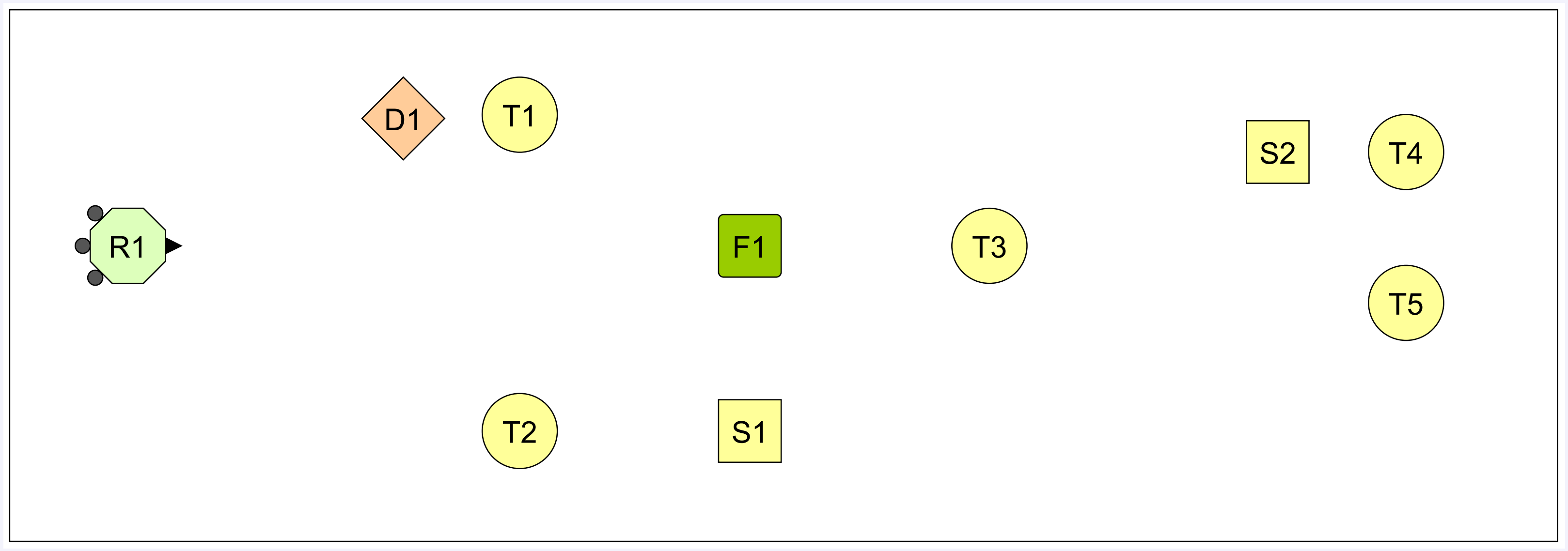
Continuous Planning and Adaptive Control in the PUG Architecture

Pat Langley
Institute for the Study of Learning and Expertise
Palo Alto, California, USA

This research was supported by ONR Grant N00014-15-1-2517. This talk reports work done jointly with Dongkyu Choi, Edward Katz, Mike Barley, and Ben Meadows.

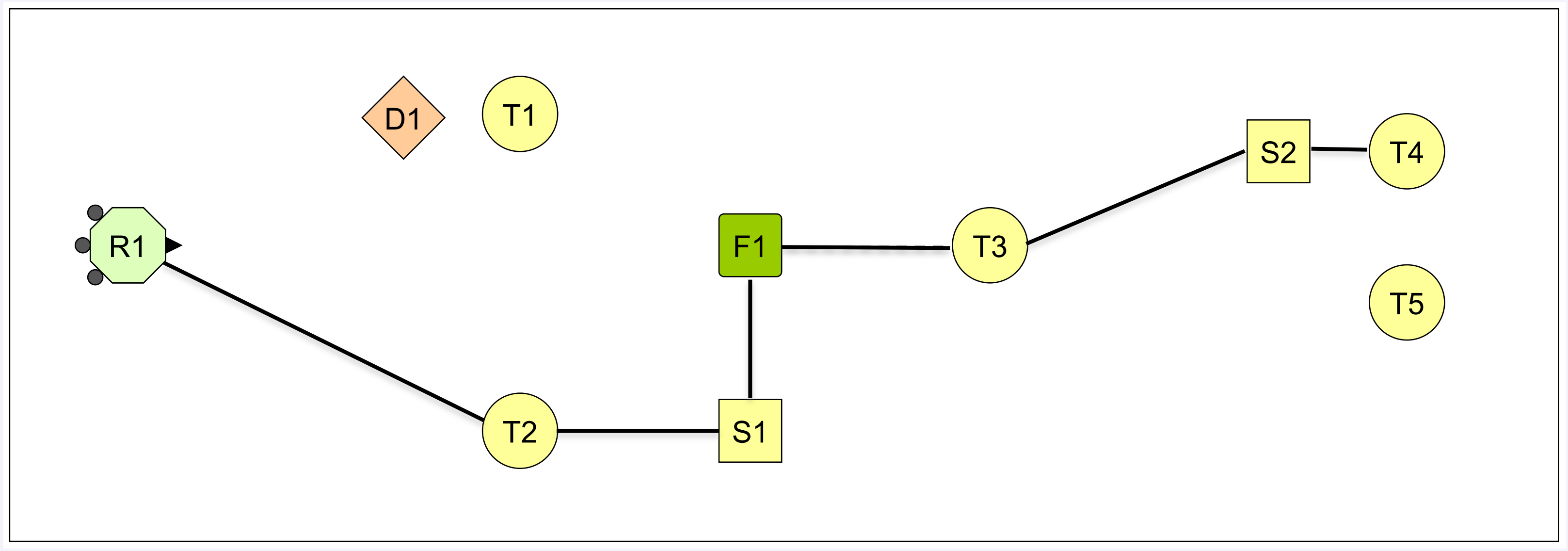
A Rover Scenario

Consider a robot scenario that includes three sensors, five target sites, one fuel depot, one sample, and one danger area.



A Rover Scenario

Consider a robot scenario that includes three sensors, five target sites, one fuel depot, one sample, and one danger area.



This task requires reasoning about *qualitative* structures and *quantitative* attributes, as well as balancing tradeoffs among *conflicting* and even *inconsistent* goals.

The PUG Architecture

We have developed a new architecture – PUG – for agents in such continuous physical domains that:

- Grounds symbolic relations in quantitative descriptions
- Associates numeric utilities with symbolic goals
- Uses numeric simulation to predict durative skills' behaviors
- Computes utilities of trajectories to guide heuristic search

PUG is a *hybrid* architecture that combines symbolic with numeric processing (Langley et al., 2016).

The framework borrows many ideas from ICARUS (Choi & Langley, 2018), but it moves beyond them.

Types of Cognitive Structures

We can organize PUG's cognitive structures into six distinct types of modular elements:

Long-Term Memories	Concepts	Motives	Skills
Short-Term Memories	Beliefs	Goals	Intentions

States, plans, and search trees organize short-term memory elements into larger configurations.

Concepts and Beliefs in PUG

Conceptual structures take the form of rules that include:

- A *relation* with one or more typed entities as arguments
- *Numeric attributes* associated with these entities
- *Conditions* that specify when the concept holds
- *Expressions* for computing derived attributes of the relation

Constituents can be defined concepts, producing a hierarchy.

A *belief* is a concept instance whose arguments are specific entities; a *state* is a set of cooccurring beliefs.

Motives and Goals in PUG

Motivational knowledge takes the form of rules that include:

- A known *concept name* and entities it relates
- *Conditions* on when that relation is desired or undesired
- The relation's *utility* as a function of numeric attributes

Motives do not define concepts; they specify the value of goals under certain conditions.

A *goal* is a motive instance with specific entities as arguments and a particular utility (which can change over time).

Skills and Intentions in PUG

PUG encodes knowledge of activities as *durative skills* with:

- A skill *name* and a set of typed entities as arguments
- *Numeric attributes* associated with these entities
- *Continuation conditions* stated in terms of known concepts
- *Executable actions* and their arguments
- *Difference equations* that predict changes caused by actions
- *Termination conditions* stated in terms of known concepts

An *intention* is a skill instance whose arguments are specific entities; it may also include a trajectory through state space.

Intentions are organized into *plans* embedded in *search trees*.

Levels of Processing in PUG

The PUG architecture involves three different processing levels, each with its own cognitive cycle:

- Inferring beliefs, goals, and utilities
- Mentally executing candidate intentions and trajectories
- Carrying out heuristic search through a space of plans

Planning builds on mental execution, which in turn builds on inference, much as in ICARUS (Choi & Langley, 2018).

An extension, PUG/X, supports plan execution, monitoring, and replanning on detecting anomalies (Langley et al., 2016).

Inferring Beliefs, Goals, and Utilities

When PUG encounters a novel state during planning, it:

- Matches concepts to generate beliefs with derived attributes
- Matches motives against these beliefs to generate goals
- Calculates the utility for each belief that satisfies a goal

When a negated goal is satisfied, it provides negative utility.

The utility of a state is the *sum* of utilities for beliefs that make up that state.

Mental Execution of Intentions

PUG determines the trajectory and utility for an intention I by:

- Applying I's actions repeatedly, starting from current state
- Using I's difference equations to predict the next state
- Deriving the beliefs, goals, and utility for this state

This continues until entering a state that matches termination conditions or fails to match continuation conditions.

The intention's utility is the *average utility* of the states that arise in its trajectory.

Heuristic Search for Partial Plans

PUG pursues forward search through a space of partial plans:

- Finding each intention that is applicable in a state
- Using numeric simulation to predict its trajectory
- Calculating the average utility of states in that trajectory
- Selecting the best untried option to extend the partial plan

Motive-linked utilities guide heuristic depth-first search; plans need *not* satisfy all goals.

Limitations of PUG

The current architecture is limited in that durative skills cannot:

- Take a goal / target state into account when taking action
  - E.g., change course when a target object changes position
- Modulate a skill's behavior to reflect other factors
  - E.g., move around obstacles on the way to a target object
- Carry out more than one skill at the same time
  - E.g., moving toward an object, turn head to watch another
- Consider natural processes when deciding on actions
  - E.g., adjust acceleration when going up or down a hill

Research on continuous control suggests promising extensions.

Three Elements of Continuous Control

Research on continuous control and robotic motion planning refers to three types of numeric attributes:

- State variables, which describe the current situation
  - E.g., a robot’s position, orientation, linear / angular velocities
- Target variables, which describe a desired situation
  - E.g., robot should be within one foot of object O and facing it
- Control variables, which can affect the situation
  - E.g., a car might set its acceleration and turning angle.

We should include these distinctions in PUG, but associate them with its cognitive structures.

Four Functions in Continuous Control

Continuous control methods relate variables using four types of equations / functions:

- Control equations [$c = f_c(s, t)$], where control variables are functions of state and target variables.
- Predictive equations [$s' = f_d(s, c)$], where derivatives of state variables are functions of state and control variables.
- Update functions [$s' = f_u(s, t)$], where derivatives of state variable depend on state, target variables (e.g., potential fields)
- Utility functions [$v = f_v(s, t)$], where values / utilities depend on state variables (e.g., reinforcement learning).

Here s is the state vector, t is the target vector, c is the control vector, and s' is the derivative of the state vector.

Two Paradigms for Continuous Control

The literature includes two paradigms for continuous control.

'Classic' control approaches:

- Control equations calculate a control vector
- Predictive equation predict resulting state (optional)
- Repeat until current state approximates target state

Potential-field approaches:

- Update function for each field computes change to state vector
- Calculate the sum of these vectors to produce a new state
- Invoke control equations to determine control vector (optional)
- Repeat until desired change in state approximates zero

Both approaches support motion planning and adaptive control.

Embedding Control Ideas in PUG

How can we incorporate ideas from continuous control into the PUG architecture? They have implications for:

- Concepts and beliefs
- Motives and goals
- Skills and intentions
- Processes for motion planning

We also want to keep classic ideas like *sequential plans* and the distinction between *means* and *ends*.

Graded Concepts and Beliefs

The new framework supports *graded concepts*, which include:

- A *relation* with one or more typed entities as arguments
- *Numeric attributes* associated with these entities
- *Expressions* for computing derived attributes of the relation
- *Means* and *variances* for each derived numeric attribute

The last specify distributions of values for derived attributes, giving a prototype for each concept.

A *belief* is a concept instance with specific entities as arguments and an associated *degree of match*.

Motives and Utilities

In the new framework, motives take nearly the same form:

- A known *concept name* and entities it relates
- *Conditions* on when that relation is desired or undesired
- The relation's *utility* as a function of numeric attributes

However, there are two important differences:

- The desired relation is stated as a single *graded* concept
- Utility considers *difference* between current and target state

Thus, the utility function is similar to a potential field but still produces a scalar value.

Skills and Intentions

Durative skills in the extended PUG architecture incorporate:

- A skill *name* and typed arguments with numeric *attributes*
- *Continuation conditions* stated in terms of graded concepts
- *Control equations* that specify control values as functions of *difference* between the current and target state
- *Difference equations* that predict changes caused by actions
- *Target / termination criteria* stated as a single graded concept

Control equations allow fine-grained, adaptive response that drives behavior toward (or away from) the target concept.

The target concept specifies a desired (or undesired) point in state space that may or may not map onto a motive.

Natural Processes

The updated architecture also supports *natural processes* that take place automatically.

- E.g., an object will fall to ground or roll down a hill.

These are similar to skills but have a simpler structure:

- A process name and typed arguments with numeric attributes
- Continuation conditions stated in terms of graded concepts
- Difference equations that predict changes from state variables

These lack control equations and target / termination criteria.

Processes are not *teleological*, although they may have a form similar to potential fields.

Mental Skill Execution / Motion Planning

Within a single skill, motion planning operates in cycles that:

- Finds the difference between the current state and target
- Uses the control equations to calculate control values
- Uses the difference equations to compute changes to the state
- Continues until target is reached or conditions not satisfied

This generates a trajectory through the state space that serves as a continuous motion plan.

Here the skill *behaves like a potential field* that is only active during its operation.

Parallel Application of Skills

We also want PUG to support parallel skill application by:

- Calculating control values as vector sum of control equations
- Computing state changes as vector sum of difference equations
- Deactivating skills separately when their targets are reached

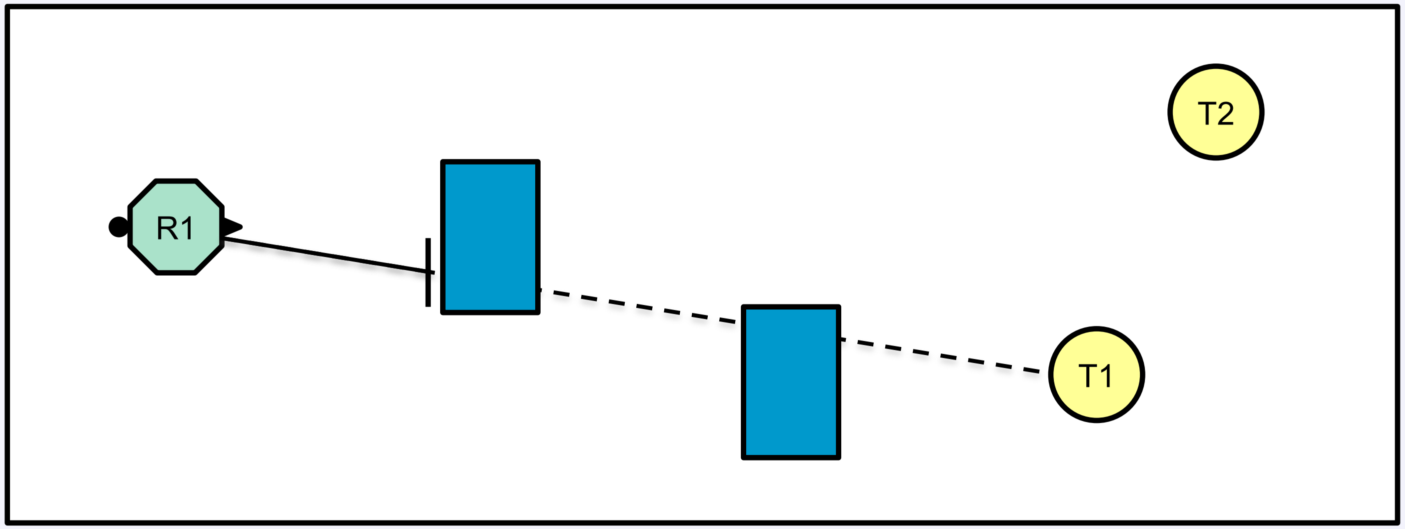
This new ability will let the architecture mimic the combined influence of multiple potential fields.

- E.g., it could mix skills for driving toward an object and avoiding an obstacle.

However, PUG would *not* apply skills with the same target concepts; one cannot move toward two targets at once.

Parallel Application of Skills

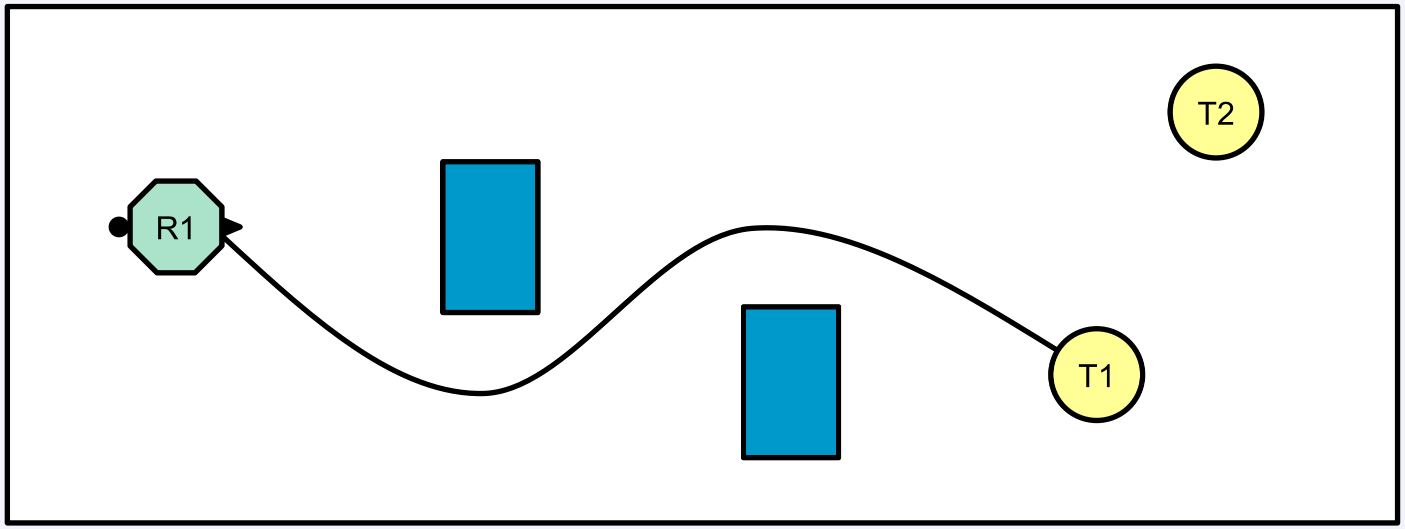
Consider a new robot scenario that includes two target sites, one sample, and two obstacles.



Because the current PUG can only apply one intention at a time and cannot adapt its behavior, it cannot reach either target.

Parallel Application of Skills

Consider a new robot scenario that includes two target sites, one sample, and two obstacles.



The extended PUG will be able to generate motion plans involving multiple intentions that influence the same control variables.

Task Planning and Heuristic Search

Task planning will remain nearly unchanged, using skills as elements for search through a problem space.

However, the extended architecture will also need to:

- Take into account the *continuous* character of states
- Incorporate *natural processes* into a trajectory when active
- Consider *combinations* of skills that are applicable

The latter ability will make candidate plans closer to *schedules* than to traditional operator sequences.

Summary Remarks

In this talk, I reviewed PUG, a cognitive architecture that:

- Encodes long-term structures for concepts, motives, and skills
- Includes short-term structures for beliefs, goals, and intentions
- Associates utilities with states, intentions, and plans

In addition, I discussed plans to incorporate ideas from classic control and potential fields:

- Graded concepts, revised motives, control equations
- Adaptive motion planning, natural processes, parallel actions

Together, these should let PUG agents exhibit a much broader range of physical behaviors.