Avoiding Local Optima with User Demonstrations

Shane Celis, Greg Hornby, and Josh Bongard

The University of Vermont

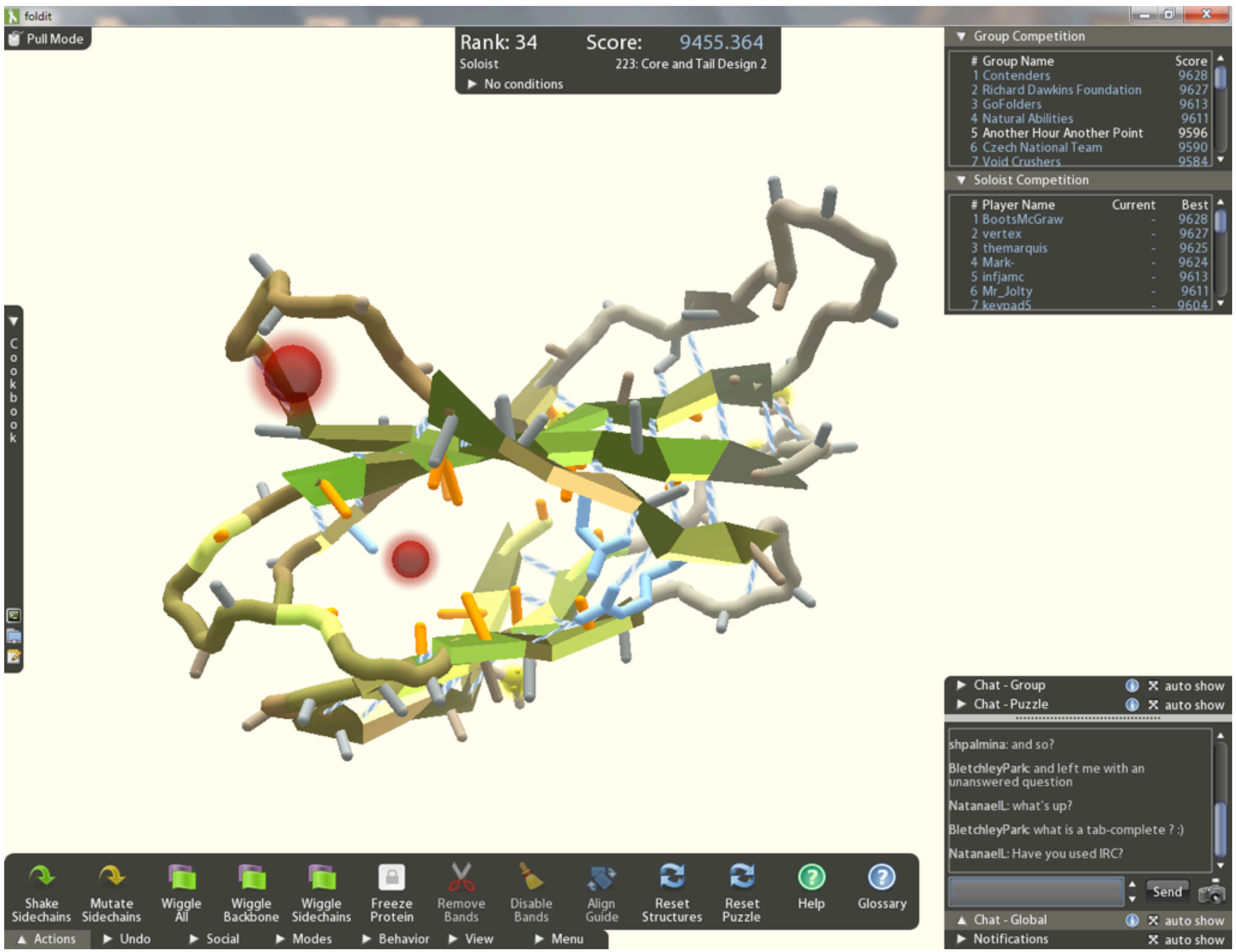
CEMS

Morphology Evolution & Cognition Lab

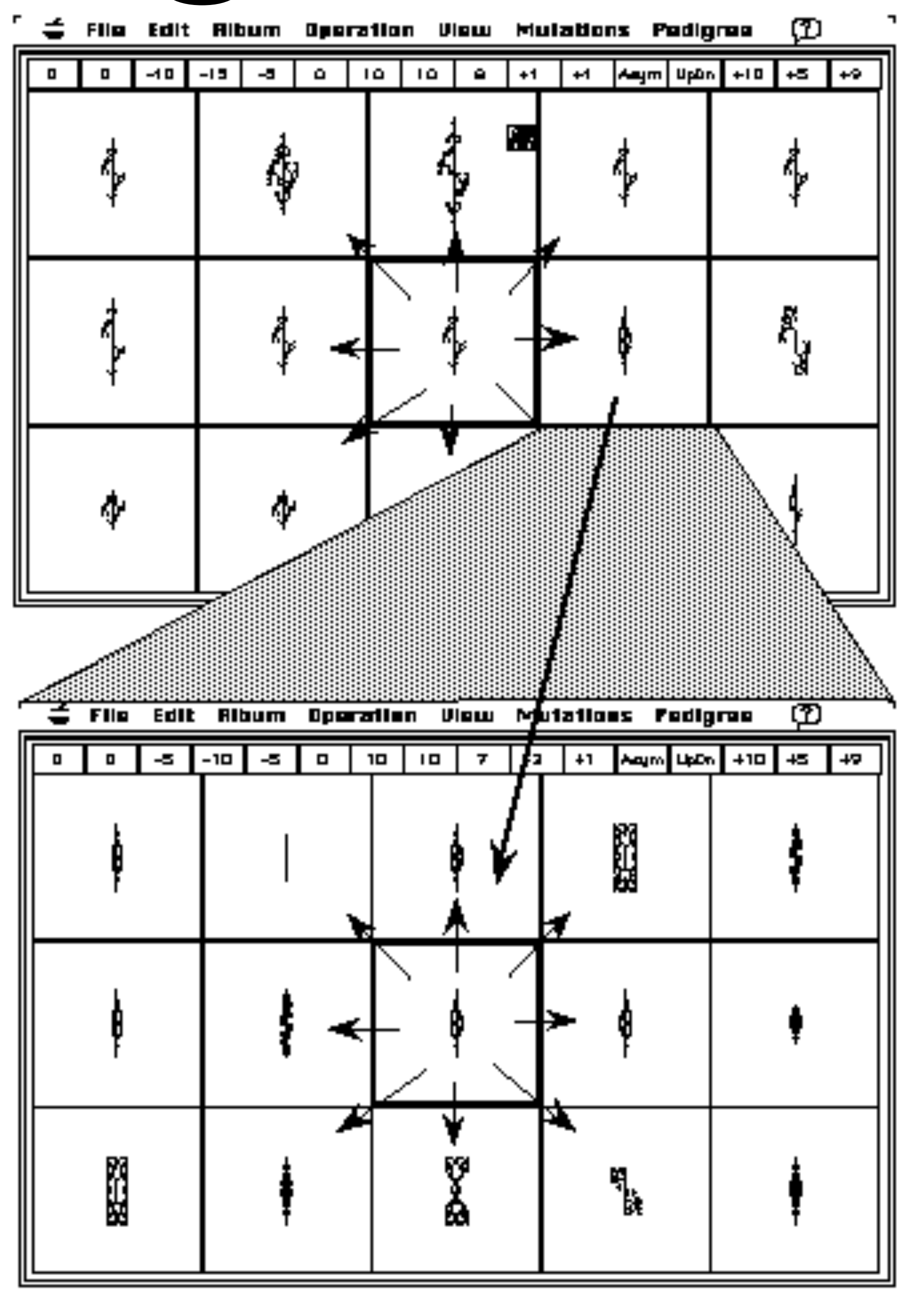
Outline

- User Guided Search
- IEAs and User Preferences
  - User Modeling
- User Demonstration
- Robot Task Environment
- High-, Mid- and Low-level Control/Fitness
- Hybrid High-level fitness with Low-level demonstration

User Guided Search



Interactive Evolutionary Algorithms



Blind Watchmaker

IEAs Guided by User Preference

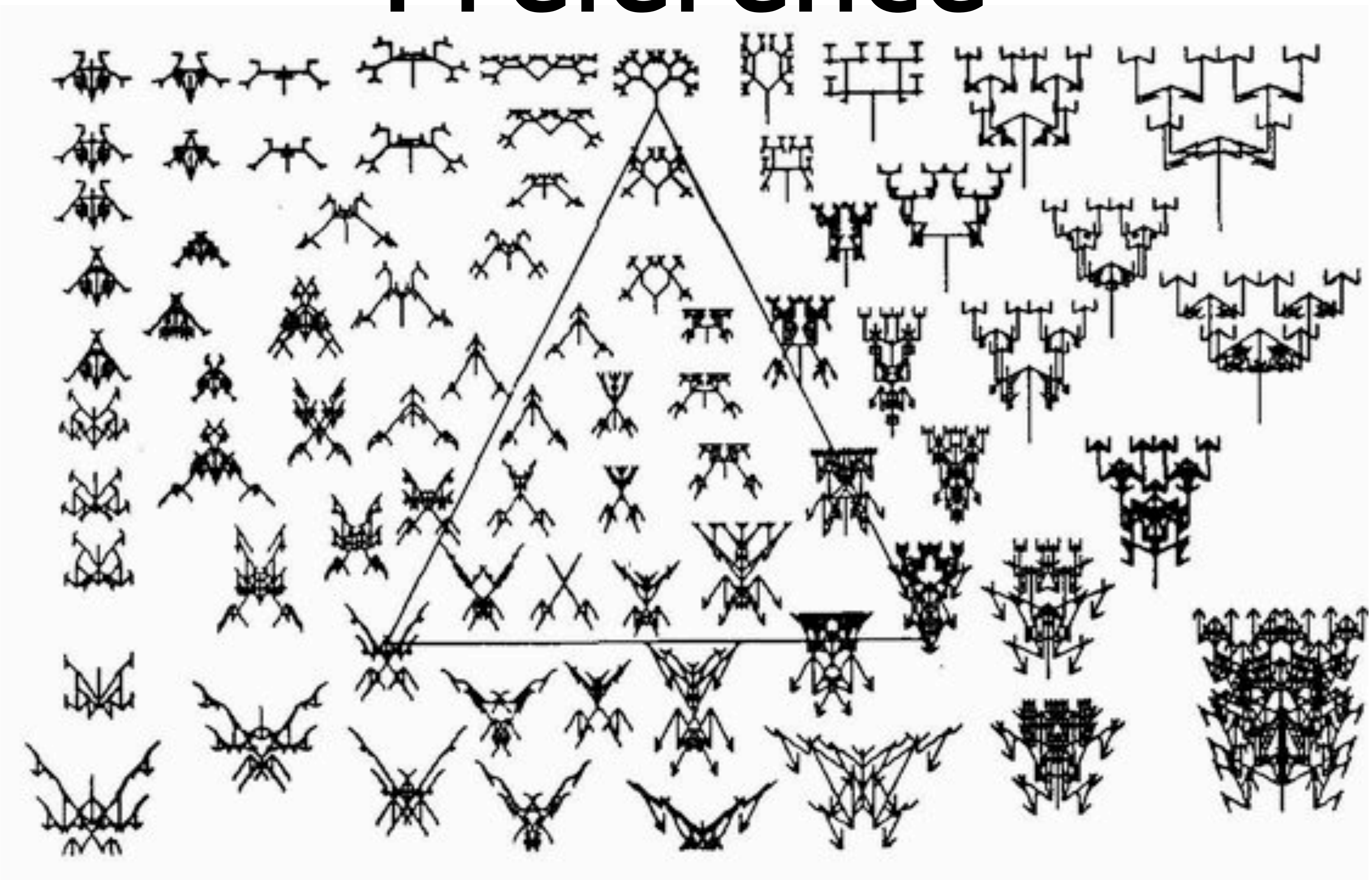
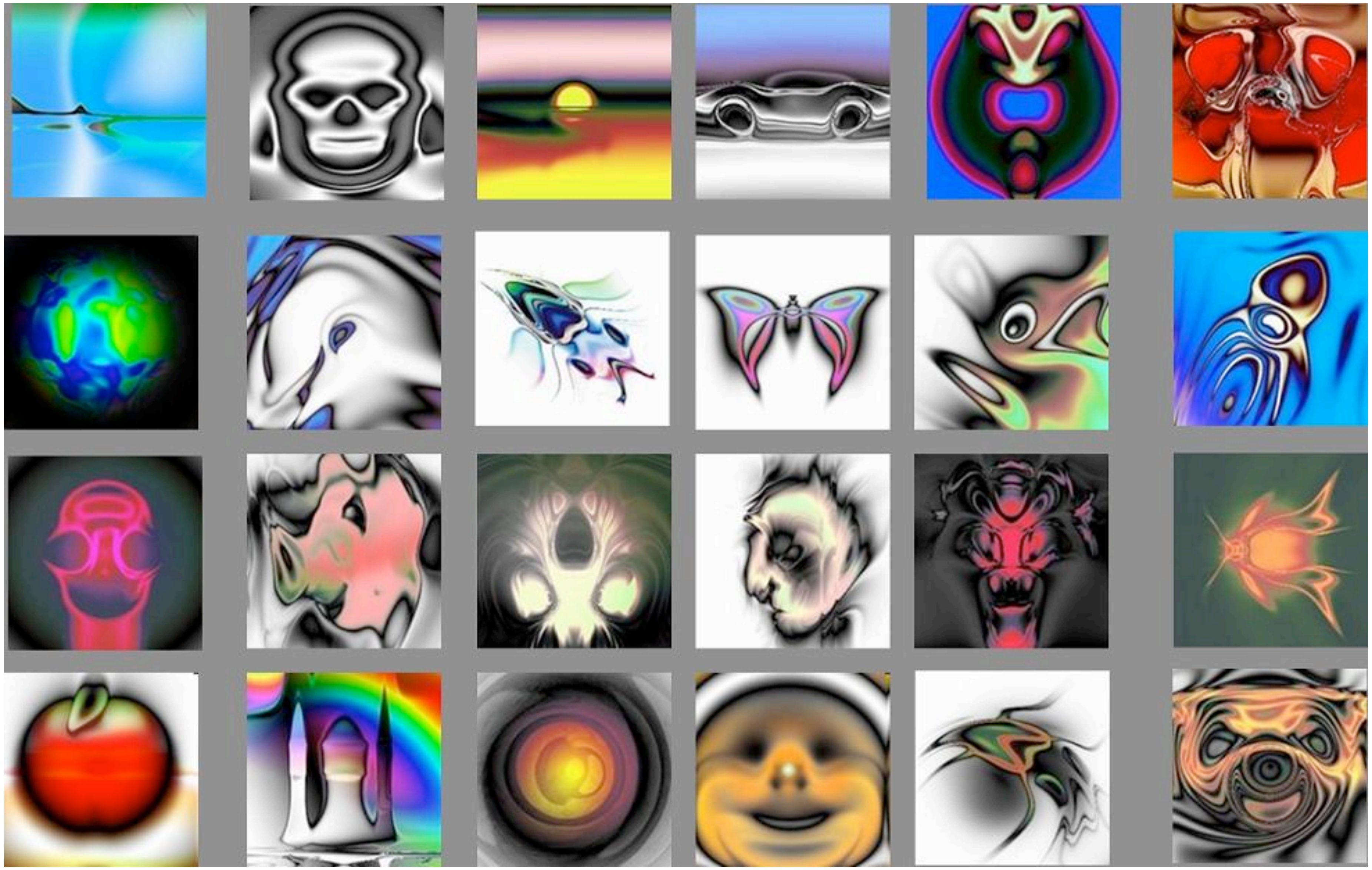
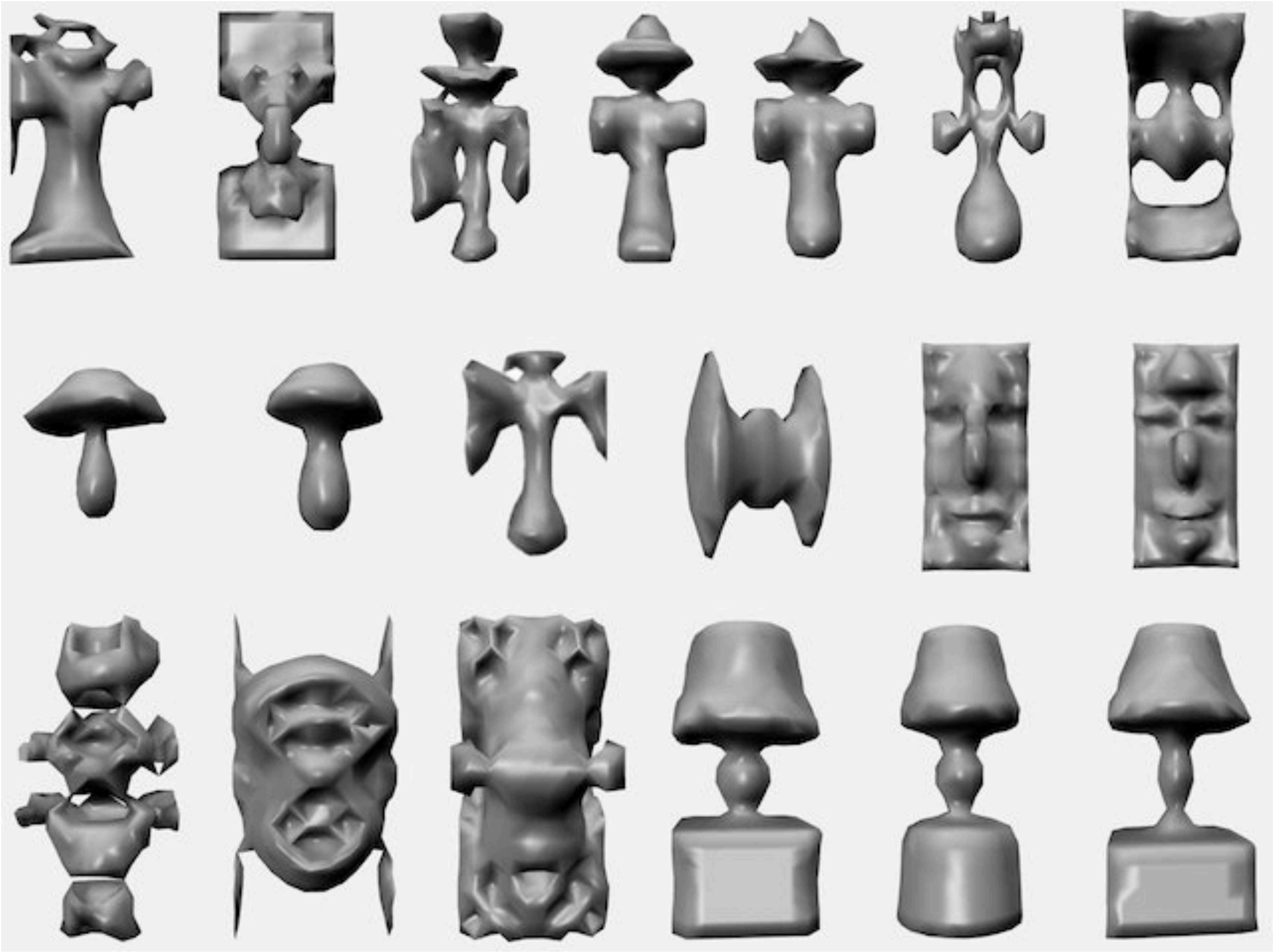


Figure 6

Picbreeder



Endless Forms



Problem

- The fitness function (human) is costly, degrades over time, and is imprecise.
- This is known as *user fatigue*.

User Fatigue

- How many evaluations are required to reach satisfactory solution?
- Non-interactive evolutionary algorithms often require thousands of evaluations.

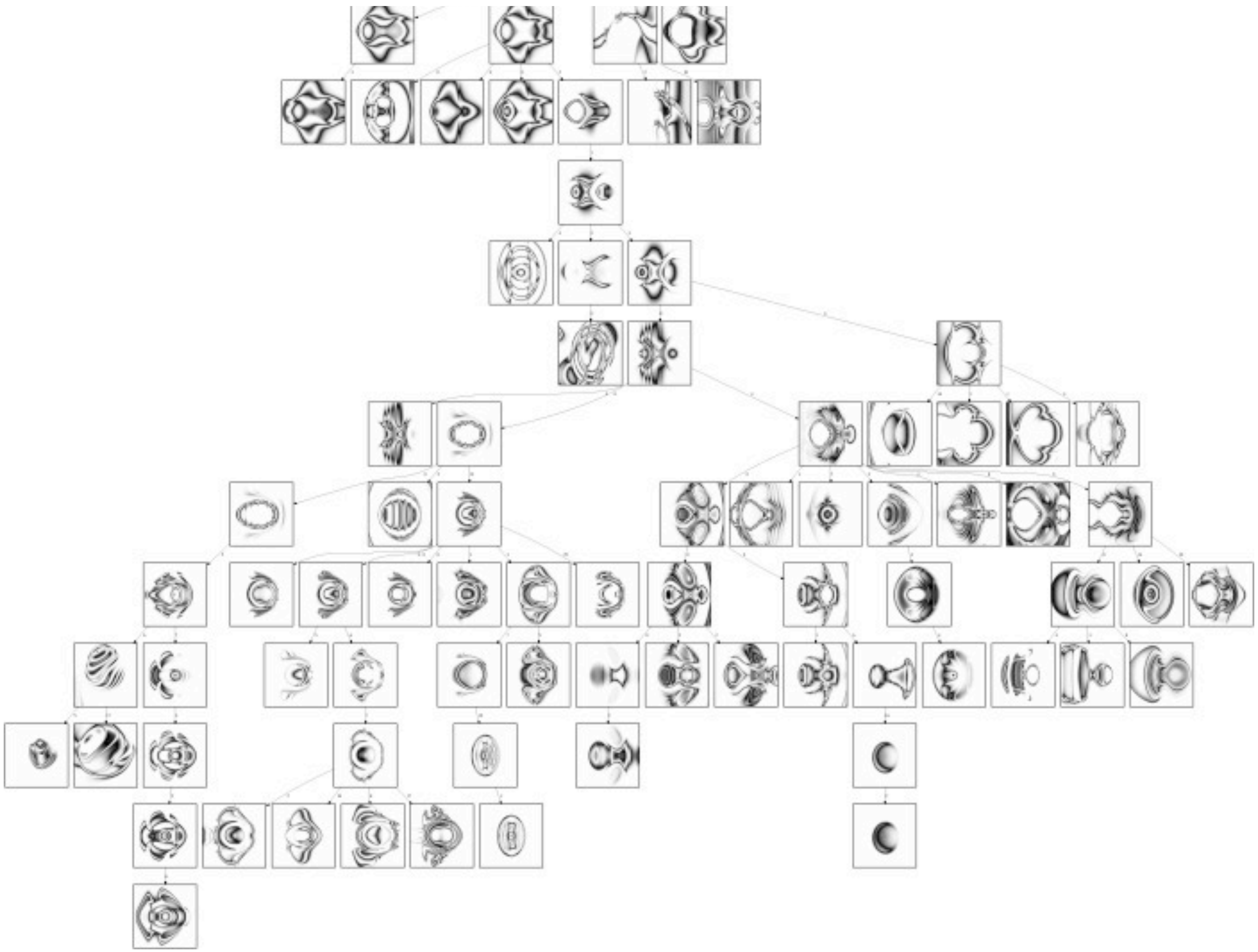
Preferences Example

- Robot Obstacle Avoidance Task
  - <u>just fitness</u>
  - <u>fitness and user preferences</u> (~200 user preference evaluations)

Dealing with User Fatigue

- Don’t require many evaluations

Crowdsource: Share the Pain



Picbreeder

- Crowdsourced Evaluations
- Expressive Encoding (CPPN)

User Modeling

- Schmidt and Lipson
- Infer preferences

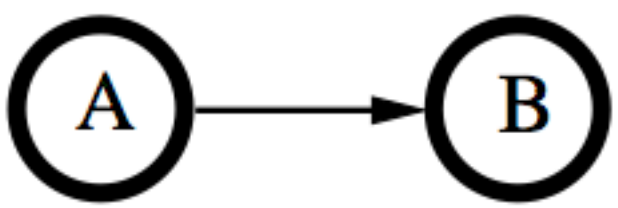
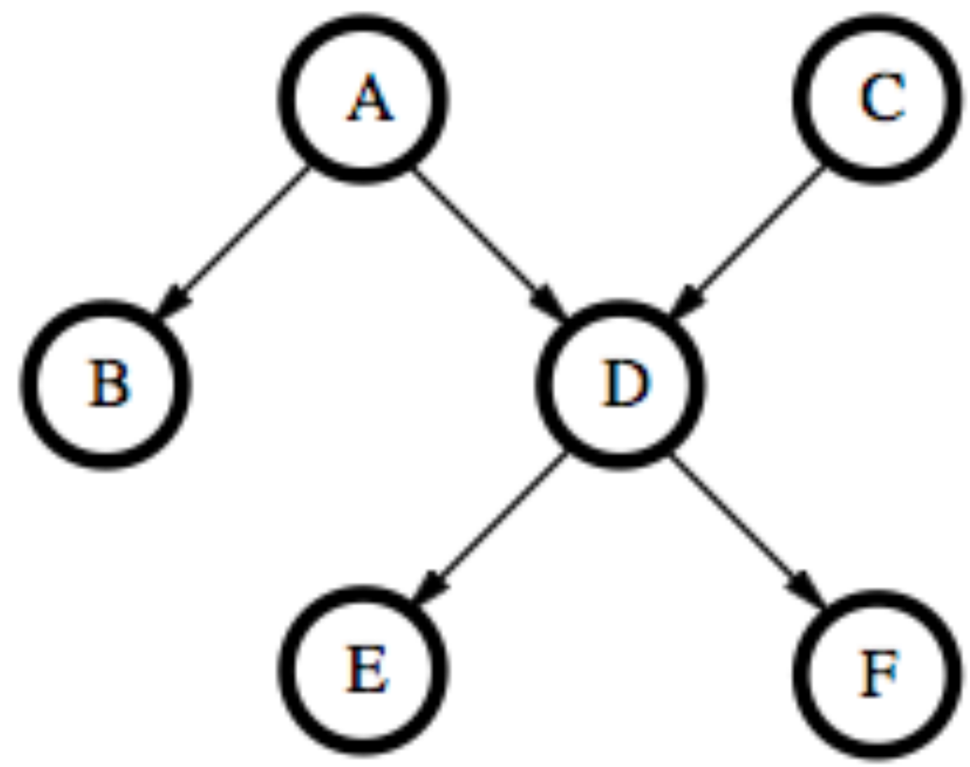
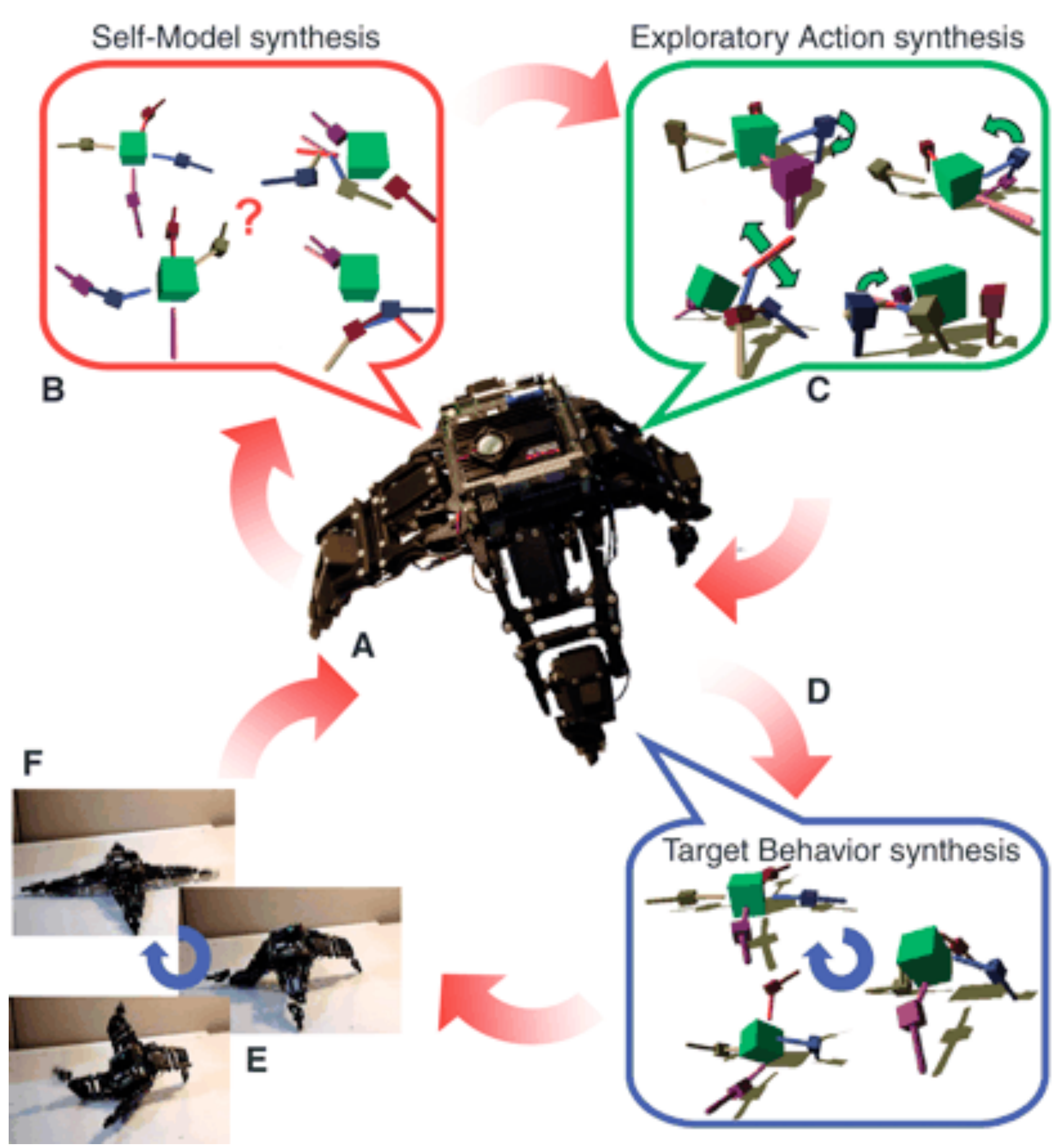


Figure 1: A simple relations graph showing that design A is preferred over design B.



Use Exploration-Estimation Algorithm of User Models



User Input

- Restricted to preferences
- User chooses between generated individuals

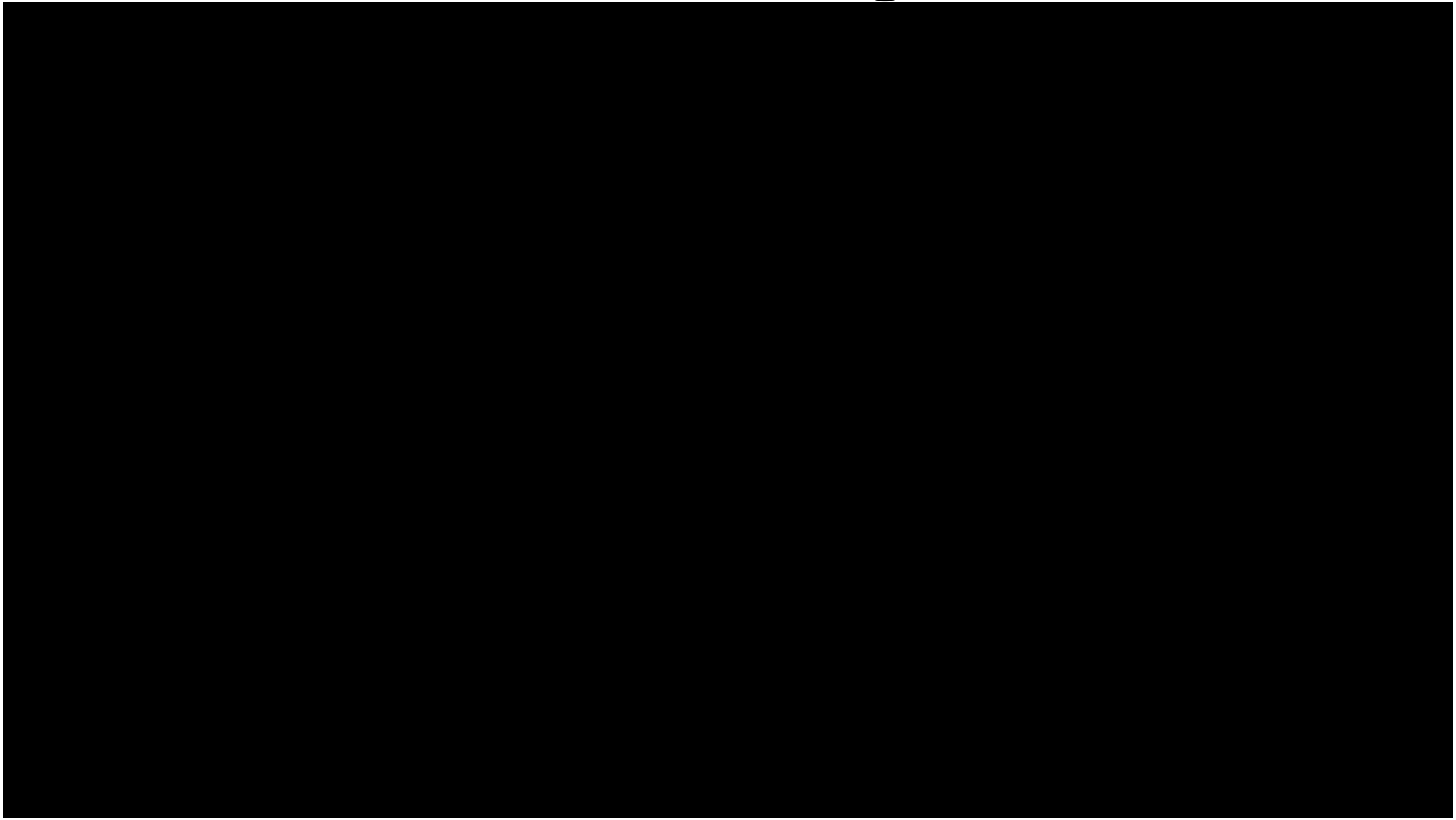
User Demonstration

- Allow the user to directly manipulate a solution

Demonstrate by Painting

Demonstrating by Molding

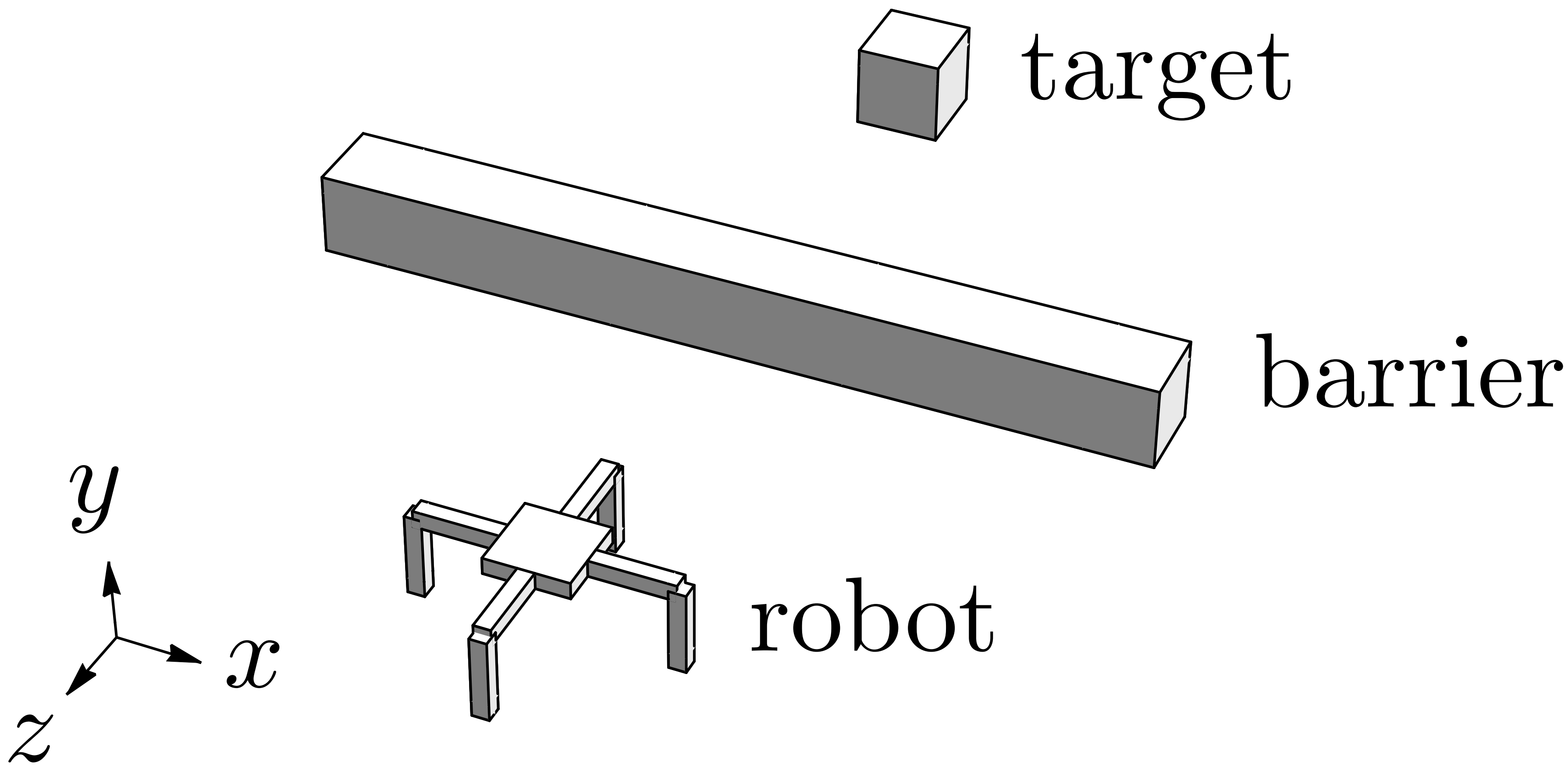
Demonstrate by Moving



(User Fatigue)^n

- Imagine having to demonstrate on every individual in a population
- Infeasible without assistance
- Must retain and reuse user demonstrations similar in spirit to how user modeling retains and reuses user preferences

Robot Task Environment



Robot

- Quadruped
- 8 degrees of freedom
  - 8 hinge joints
- 2 light sensors
- 2 time measures (fast for gait, slow for task)
- Neural network controller (4 input, 12 hidden, 12 hidden, 8 output)

High-, Mid-, and Low-level Control

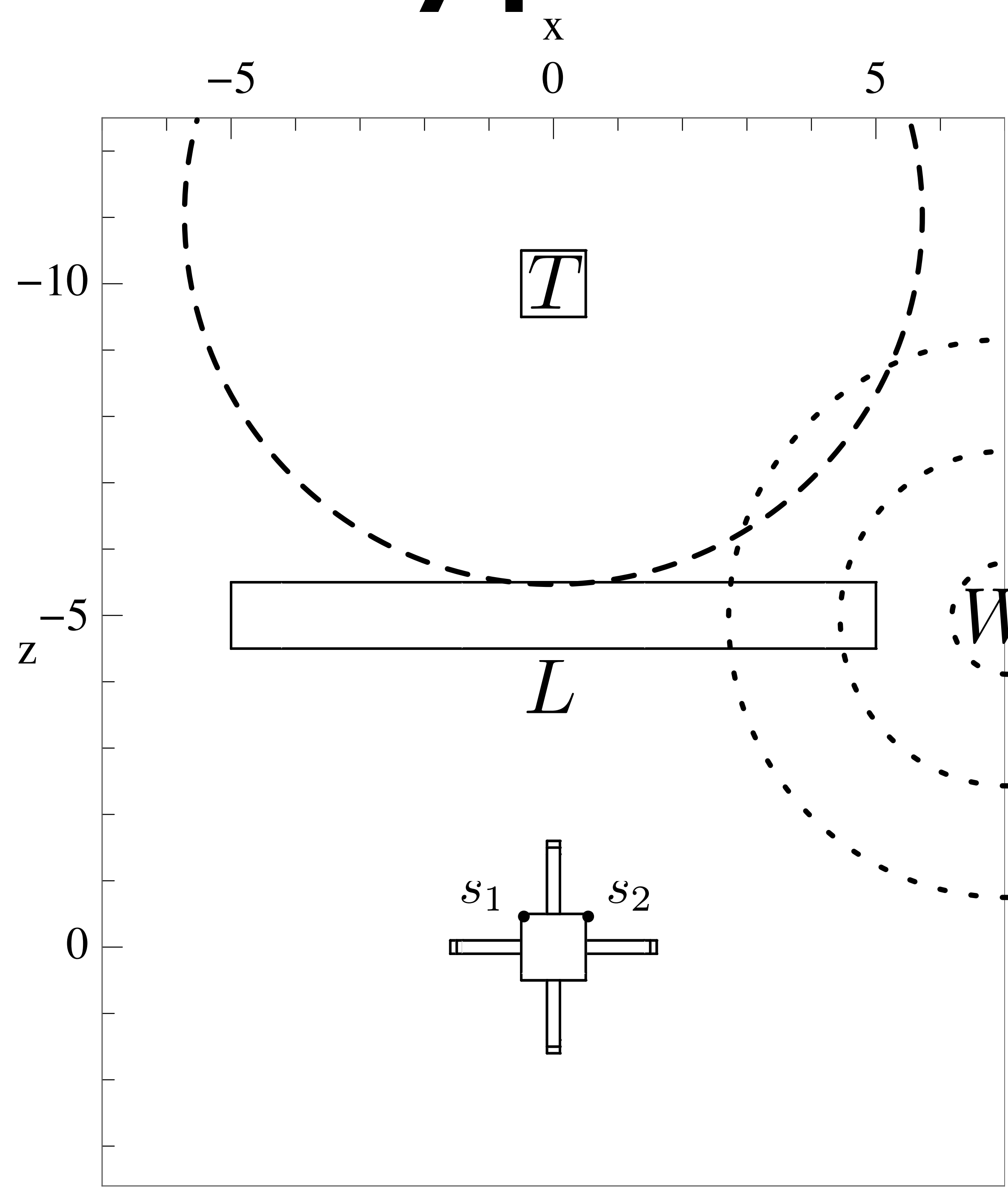
- High-level control might command the robot to go to the target.
- Mid-level control might command the robot to go right, up, left, to reach the target.
- Low-level control would command all the joint positions.

High-level Fitness

$$f_{\text{high}} = ||\mathbf{r}_r(t_f) - \mathbf{r}_t||$$

Minimize this!

Mid-level Fitness, Waypoints



Mid-level Fitness

$$f_1(t) = \frac{||\mathbf{r}_r(t) - \mathbf{r}_w||}{||\mathbf{r}_r(t_0) - \mathbf{r}_w||}$$

$$f_2(t) = \frac{||\mathbf{r}_r(t) - \mathbf{r}_t||}{||\mathbf{r}_r(t_1) - \mathbf{r}_t||}$$

$$t_1 = \min_t f_1(t) < \alpha$$

$$f_{\mathrm{mid}} = \frac{1}{t_f} \sum_{t=0}^{t_f} \begin{cases} f_1(t) & t < t_1 \\ \alpha f_2(t) & \text{otherwise} \end{cases}$$

Hybrid Fitness

$$[f_{\text{hybrid}}]_1 = f_{\text{high}} = ||\mathbf{r}_r(t_f) - \mathbf{r}_t||$$

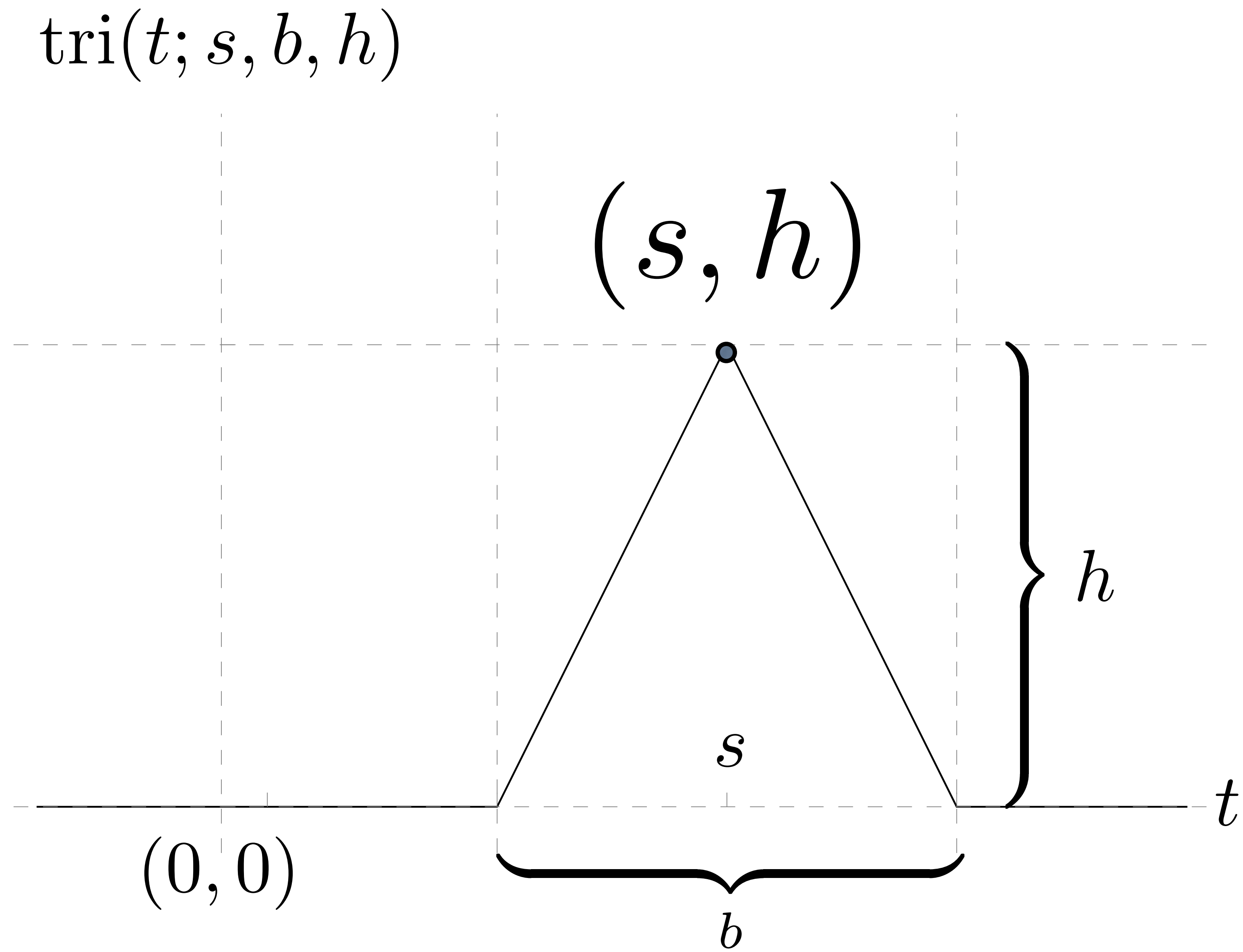
$$[f_{\text{hybrid}}]_2 = UDE$$

User Demonstration Error (UDE)

User Demonstration

- A set of tuples that each define the time, joint, and joint position (s, i, h)
- For simplicity, let's pretend the user only provides one demonstration value.
- Because this interacts with a continuous system, we want to smooth it somehow.

Smooth the User Demonstration



Construct a New Controller

- Given a prior controller theta(t), construct a new controller that satisfies the user demonstration.

$$\theta(t)' = \theta(t) + \mathrm{tri}(t; s, b_c, h - \theta(s))$$

User Demonstration Error (UDE)

- Three driving considerations:
  1. When the user demonstrates *h* at time *s*, that should be the maximum error (wrt that demonstration).
  2. When the user has performed no demonstration near time *s*, there should be no error.
  3. In between those extremes, use an intermediate value.

User Demonstration Error at Time *t*

- Determine absolute difference between prior controller and the constructed controller
- Only accept differences near the user demonstrations.

$$ude(t) = |\theta(t)' - \theta(t)| \, \mathrm{tri}(t; s, b_e, 1)$$

Add it up!

$$UDE = \int_0^{t_f} ude(t)\,\mathrm{d}t \approx \sum_{i=0}^{m} ude(i\,\Delta t)$$

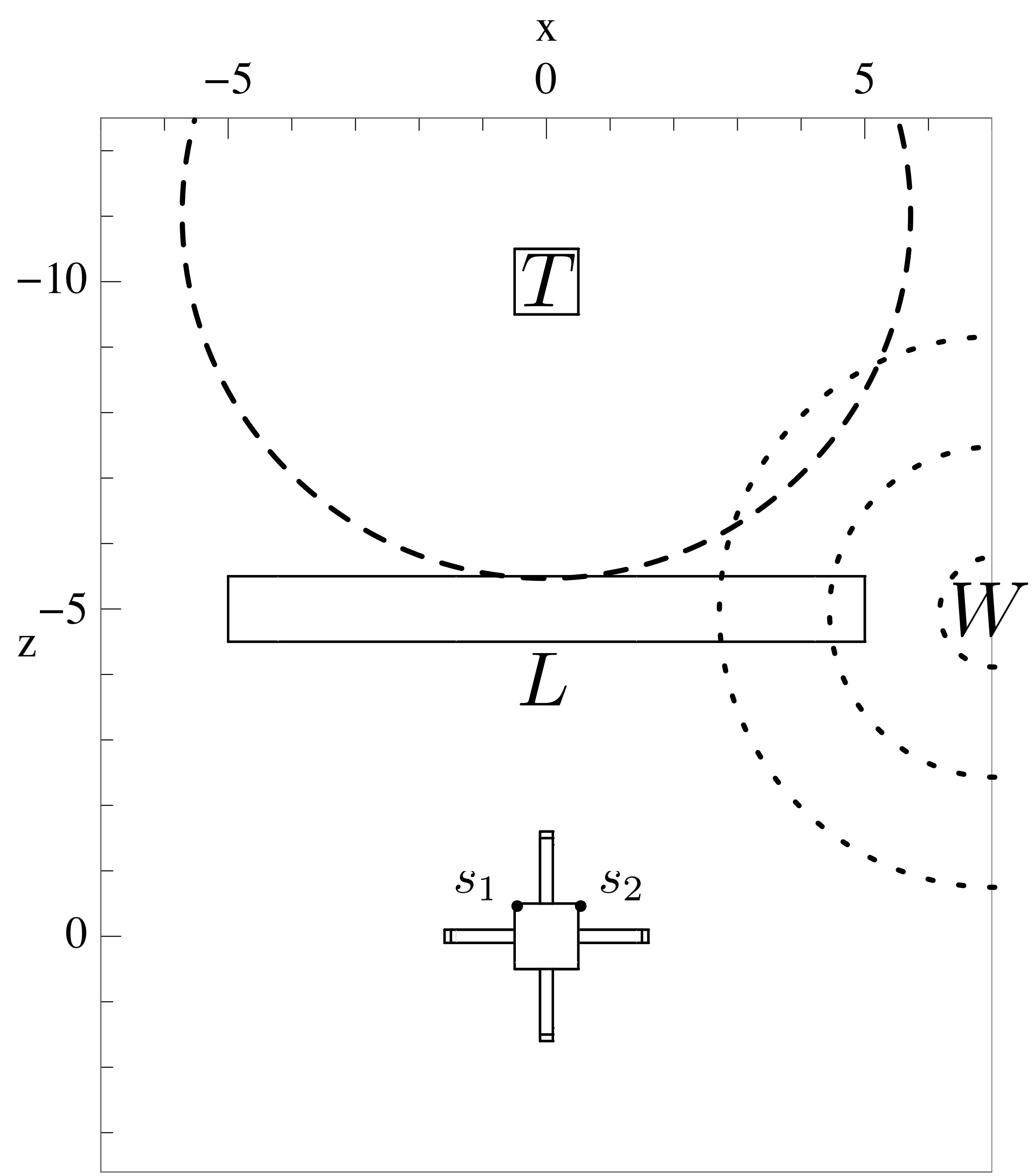
Hybrid Fitness Refresher

$$[f_{\text{hybrid}}]_1 = f_{\text{high}} = ||\mathbf{r}_r(t_f) - \mathbf{r}_t||$$

$$[f_{\text{hybrid}}]_2 = UDE$$

User Demonstration Error (UDE)

Refresh on Task



Surrogate User

- Using the system interactively, one can determine how to move the robot in a cardinal direction.
- The surrogate sets up a oscillating motion that propels the robot to the right.
- Is this cheating? No, the user is guiding the search with low-level input.

Experiments

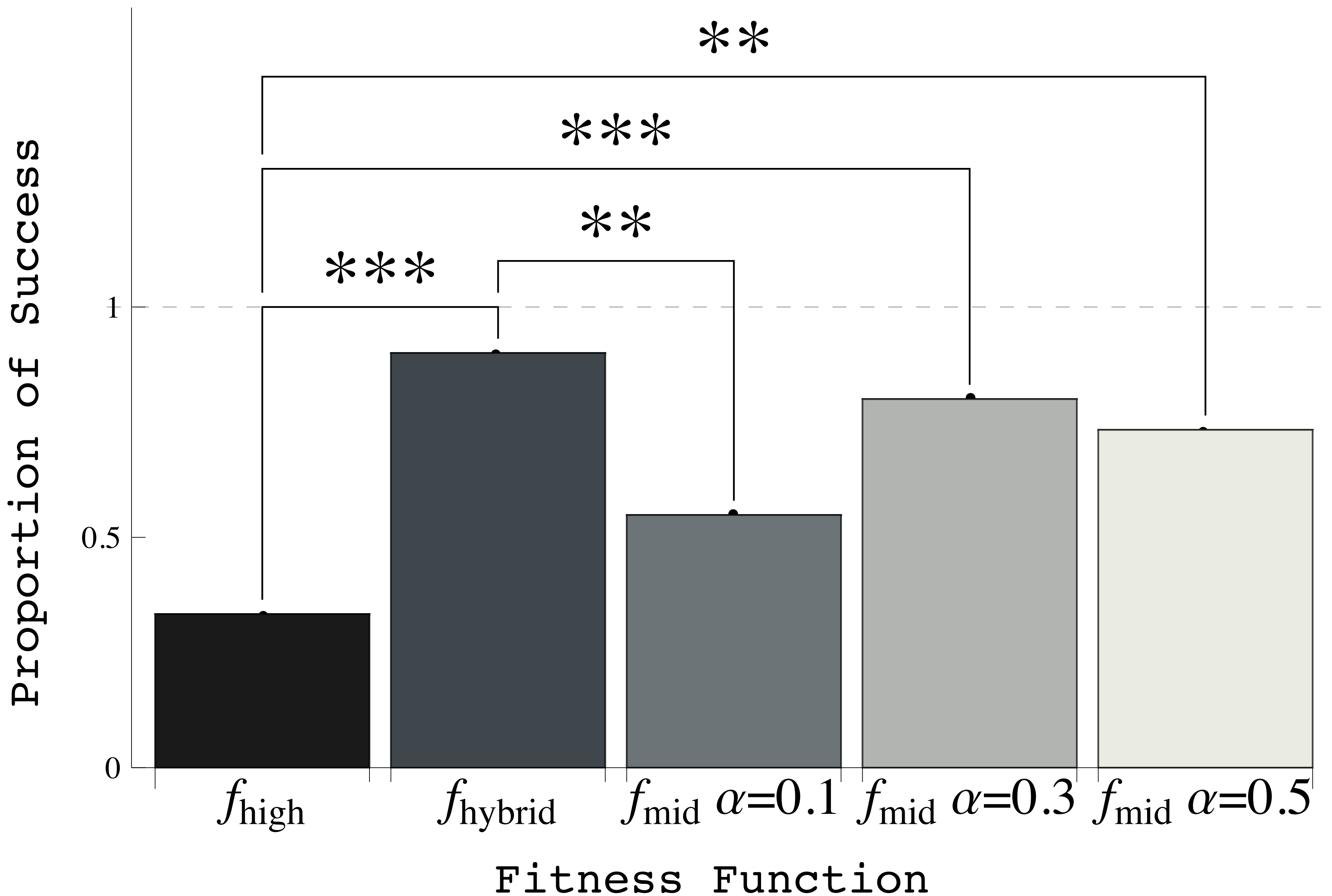
- 30 independent trials for each fitness functions: f_{high}, f_{mid}, and f_{hybrid} (3 parameter settings),
- NSGA-II used with population of 20 for 100 generations.
- Success defined as reaching within 4.5 units of the target object.

Table Results

Experiment	Percent Successful	p-value
f_{high}	33.3%	$p < 0.001$
f_{hybrid}	90.0%	$p = 1$
f_{mid} $\alpha = 0.1$	54.8%	$p < 0.01$
f_{mid} $\alpha = 0.3$	80.0%	$p = 0.5$
f_{mid} $\alpha = 0.5$	73.3%	$p = 0.2$

- P-values shown are compared with f_{hybrid} using the Exact Fischer Test.

Results


**
***
**
***
1
0.5
0
Proportion of Success
f_{high}
f_{hybrid}
f_{mid} α=0.1
f_{mid} α=0.3
f_{mid} α=0.5
Fitness Function

Conclusion

- Compared a system that accepts low-level user demonstrations coupled with a high-level fitness function
- overcomes a local optimum
- addresses the user fatigue problem with user demonstration error (UDE)
- suggests low-level, inexpert demonstrations may be a good way to guide search

Future Work

- Test with humans
- Test with an interactive user surrogate
- Test with a different task environment, e.g., a jump task

Thank you for your time.

Questions?