AI ARTIFICIAL INTELLIGENCE SUMMIT

Beyond Bots: Making Machine Learning Accessible and Useful

Joe Booth - Vidya Gamer, LLC

Wolff Dobson - Google Inc

Danny Lange - Unity

"There is ***no single development****, in either technology or management technique, which by itself* ***promises*** *even* ***one order-of-magnitude improvement*** *within a decade in productivity, in reliability, in simplicity."*

Fred Brooks

'No Silver Bullet' 1986

gradient-free methods
(e.g. NES, CMA, etc.)

10x

fully online methods
(e.g. A3C)

10x

policy gradient methods
(e.g. TRPO)

10x

replay buffer value estimation methods
(Q-learning, DDPG, NAF, etc.)

10x

model-based deep RL
(e.g. guided policy search)

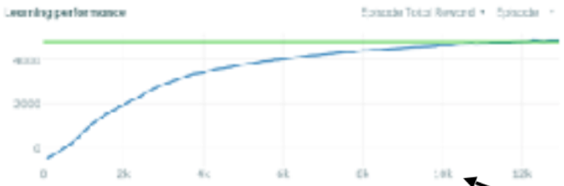
10x

model-based "shallow" RL
(e.g. PILCO)

Evolution Strategies as a Scalable Alternative to Reinforcement Learning

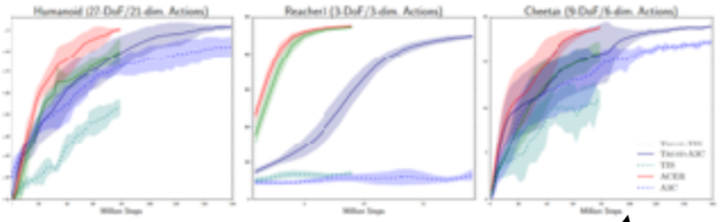
Tim Salimans[1] Jonathan Ho[1] Xi Chen[1] Ilya Sutskever[1]

half-cheetah (slightly different version)



TRPO+GAE (Schulman et al. '16)

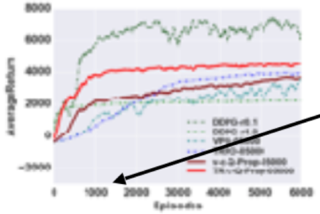
10,000,000 steps
(10,000 episodes)
(~ 1.5 days real time)



Wang et al. '17

100,000,000 steps
(100,000 episodes)
(~ 15 days real time)

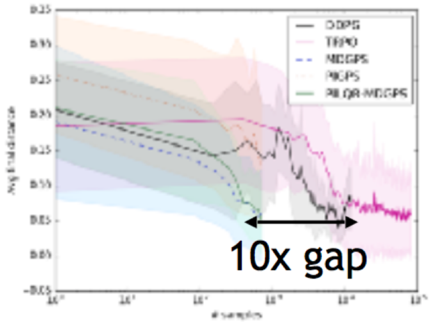
half-cheetah



Gu et al. '16

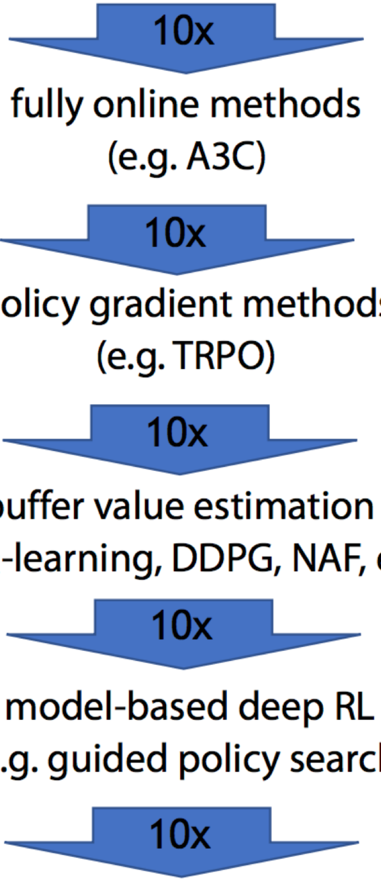
1,000,000 steps
(1,000 episodes)
(~ 3 hours real time)

	cart pole	cart-double pole	unicycle
state space	$\mathbb{R}^4$	$\mathbb{R}^6$	$\mathbb{R}^{12}$
# trials	≤ 10	20–30	≈ 20
experience	≈ 20 s	≈ 60 s–90 s	≈ 20 s–30 s
parameter space	$\mathbb{R}^{305}$	$\mathbb{R}^{1816}$	$\mathbb{R}^{28}$



Chebotar et al. '17 (note log scale)

about 20 minutes of experience on a real robot



Slide from Sergey Levine

Hypothesis:

Reinforcement Learning along with reward functions, if applied to general software development will result in a multi-magnitude improvement in productivity, in reliability, in simplicity.

Use video game AI as a proof.

my goal today

Pt1 – taste of rl

Pt2 - Share research results

Joe’s background

Softography 1986 to 2012

30+ Titles
24m Sales
$1b Revenue
10 AIAS & BAFTA nominations

Early Pioneer of *Online / Social Play*
Multi-genre:
Sports, Shooters, Platform, Driving, Simulation, Kids
Multi-Disciplines:
Production, Direction, Engineering

Impact (Neuroscience)

Augmented AI


Annie Duke


Dave Lenowitz


Eric Brooks


Dr Brock Eide


Nils Lahr

HOW I DECIDE

NEUROSCAPE

AI CITY

NATIONAL GEOSPATIAL-INTELLIGENCE AGENCY

UNITED STATES OF AMERICA

rl Research

A taste of rl

The problem with learning rl

Bootcamp Goals

- Understand mathematical and algorithmic foundations of Deep RL
- Have implemented many of the core algorithms

Prerequisites

- Probability,
- Calculus,
- Linear Algebra,
- Graphical Models
- CS Supervised Learning

$$V^*(s) = \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi, s_0 = s\right]$$

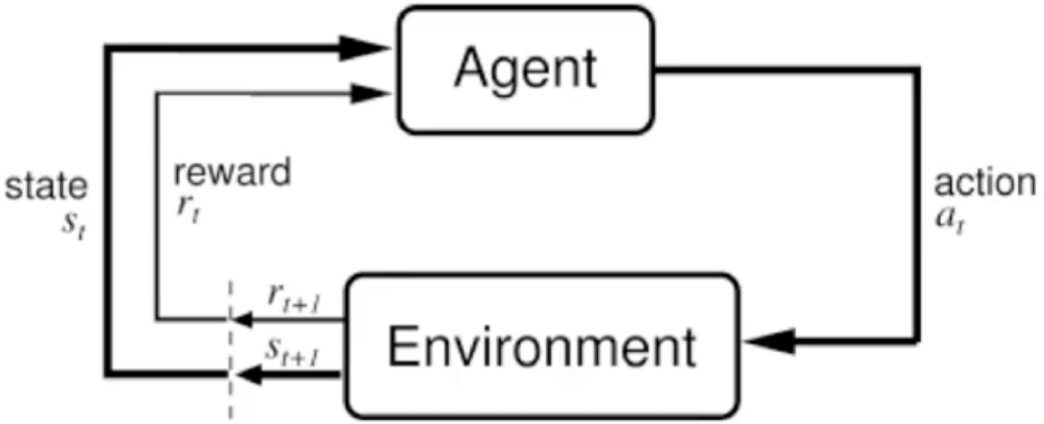
Advice

- Build mental model - gist
- Do get hands on
  - Recreate benchmarks (Classic Control, Atari, MuJoCo)
- Don’t waste time with online ml courses

Plus

- Podcasts
- Videos
- Skim read / re-read papers
- machinelearningmastery.com

Mental Model:



Assumption: agent gets to observe the state

[Drawing from Sutton and Barto, Reinforcement Learning: An Introduction, 1998]

incompleteideas.net/book/the-book.html

Mental Model:

Environment = Pixels -> CNN

Compresses problem space

Actions

Left, Right, No-Op

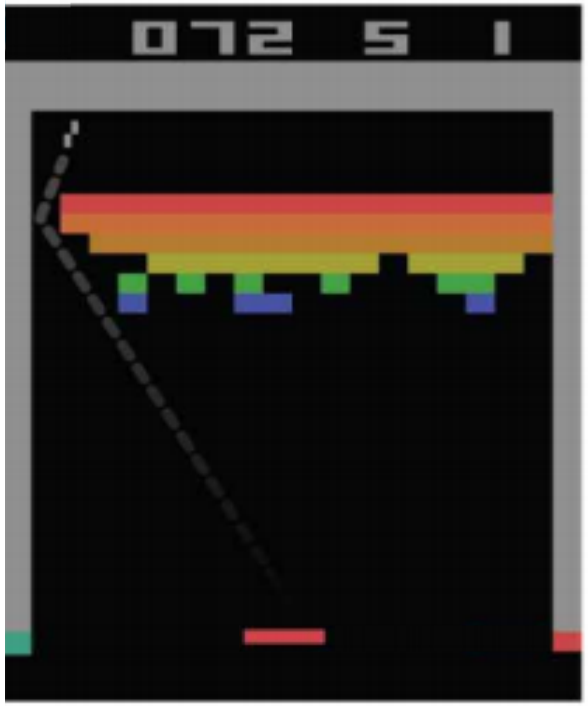
Delayed **Discounting**

step reward += (step+1 reward ***.98**)

Experience Replay

1,000,000 buffer.

Random sample during training



Mental Model:

Exploration vs Exploitation

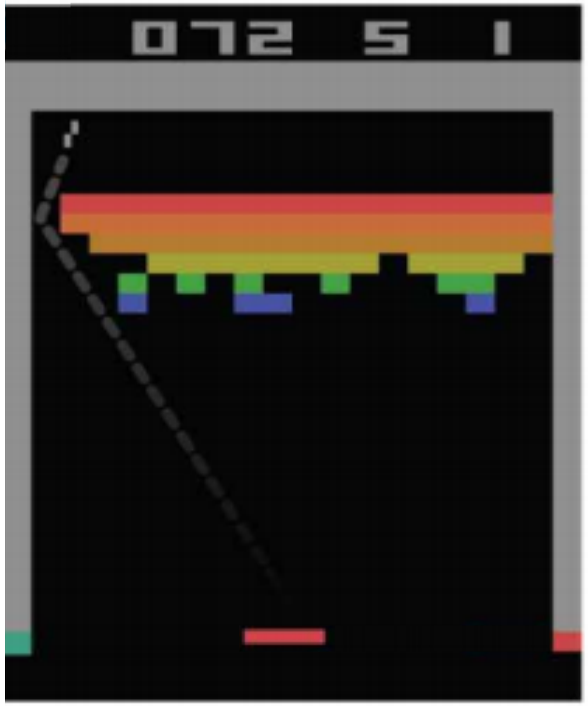
100% random to 2%

… naïve,

- 100% for first 1m steps; reduce until 2%

…on dyslexia and automaticity

- Prof Rod Nicolson
- n * sqrt(n)
- 100 = 1,000 repetitions
- 900 = 27,000 repetitions



Algorithms	vs	Frameworks
DQN (Torch)	DeepMind	DeepMind Lab PySC2 – StarCraft 2
Baselines (A2C, ACER, ACKTR, DDPG, DQN, GAIL, HER, PPO, TRPO)	OpenAI	OpenAI Gym (Classic Control, Atari, MuJoCo, Toy text, Algorithms, Box2d, Robotics) OpenAI Universe (GTA, FlashGames, Browser)
ML Agents (PPO, Behavioral Cloning)	unity	Unity Game Engine (10+ sample environments)

Hands on: Linear regression example

- Took high-cost function
- Recorded tons of data
- Trained offline
- Swapped function for model
- Compare with function

Analog (Baselines + Unity)

localhost

DashboardRx Home Evaluation Algorithm Actor ActorInstance Enviroment Experiment Problem

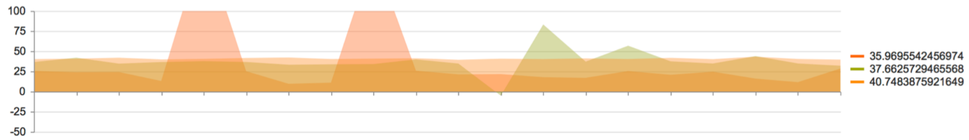
ai_dm_walker-v0-0002

Edit

Hypothsis: 200k steps / new gears / _velocityScaler run1=30, run2=40, run2=50, Edit

ActorInstance: ai_dm_walker-v0-0002 (a_dm_walker-v0, Ddpg-v0)

Params: { "nsteps":1000, "nb_epochs": 100, "max_score_episodes": 20 } { "fps":125 } Edit



Average Score: **38.13** (- 35.9695542456974 - 40.7483875921649 - 37.6625729465568 -)

Observation: scaler @ 50 looks like it is trying to skip (maybe too powerful but lets try longer run).... @40 scored the most during traing. Maybe try 45??. 200k steps is not enough to learn with Edit

CLONE ENQUEUE RUN

exp_Xq-WrDJvWUePuPwr_yDpRg

Actor instance ai_dm_walker-v0-0002 scored **35.97** (+/- 23.96) on the enviroment RagdollSnake-v0 using the algorithm Ddpg-v0

Score: 35.97 (+/- 23.96)

Ave Steps: 305.05(+/- 116.06)

Id: eval_exp_Xq-WrDJvWUePuPwr_yDpRg_score

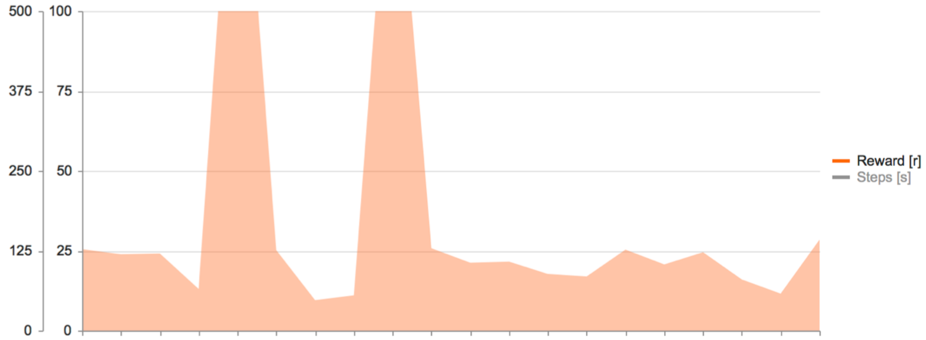
Mode: _score

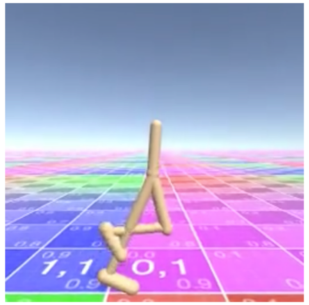
Start: 03/16/2018 17:48:41

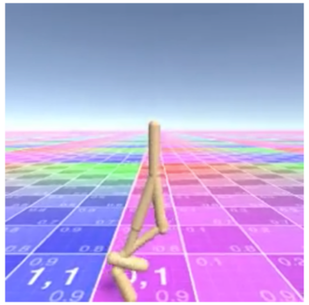
Vides: 3

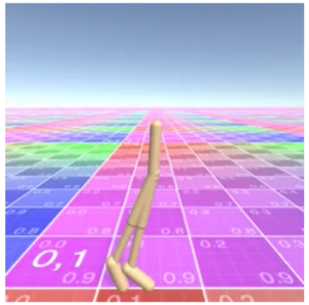
Total Episodes: 20

Scoring Evaluation









Takeaway

Baselines not really modular, has bugs, hard to maintain

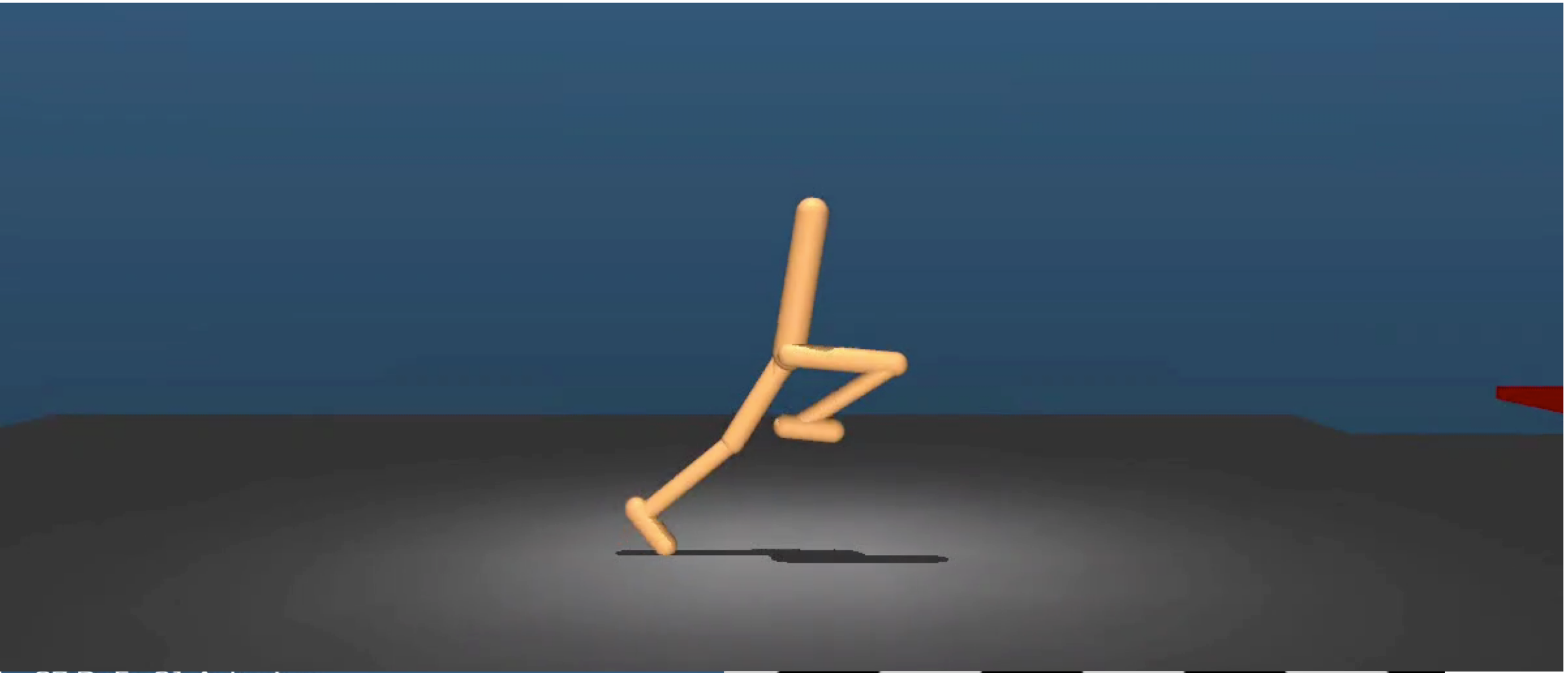
Structure really helps me

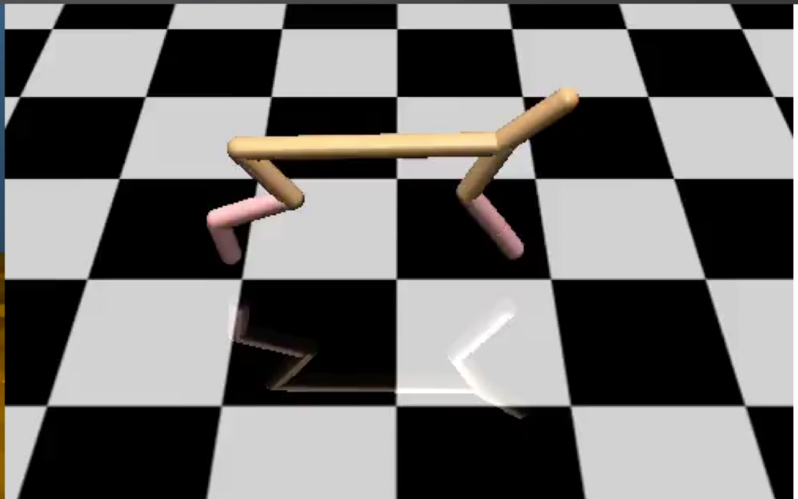
publishing results

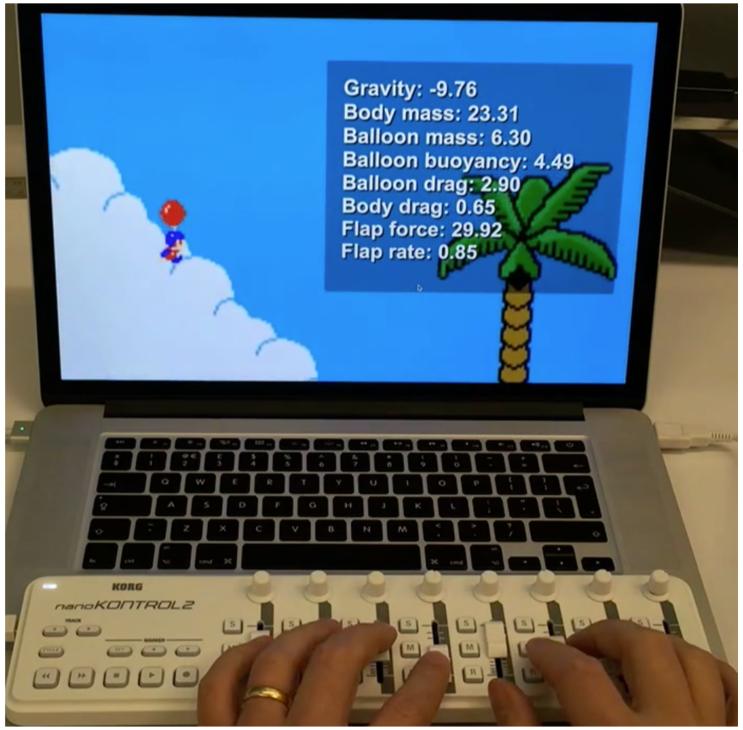
#1 **rl locomotion:**

Reproducing MuJoCo benchmarks in a modern, commercial game /physics engine

rl locomotion: abstract







GDC 2015

Designing with Physics:
by Bennett Foddy

rl locomotion: method

Architecture

- Analog
  - Game Engine: Unity + PhysX (same physics engine as Unreal)
  - Python: OpenAI.Gym + Baselines

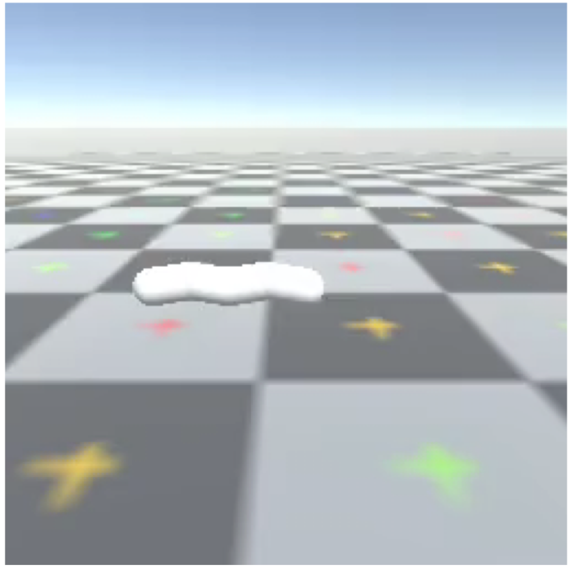
Algorithms

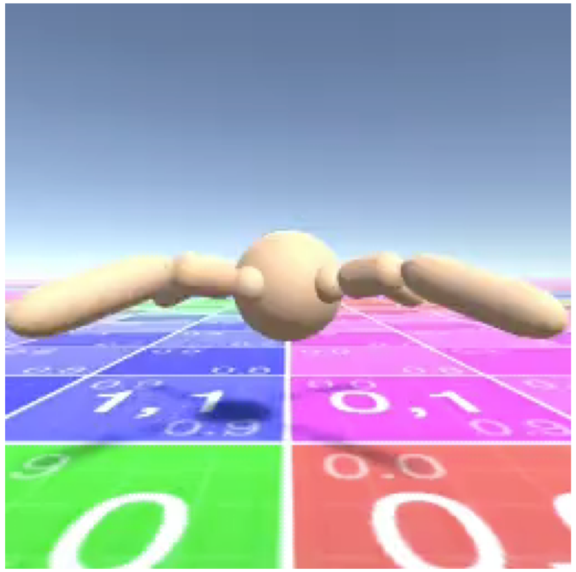
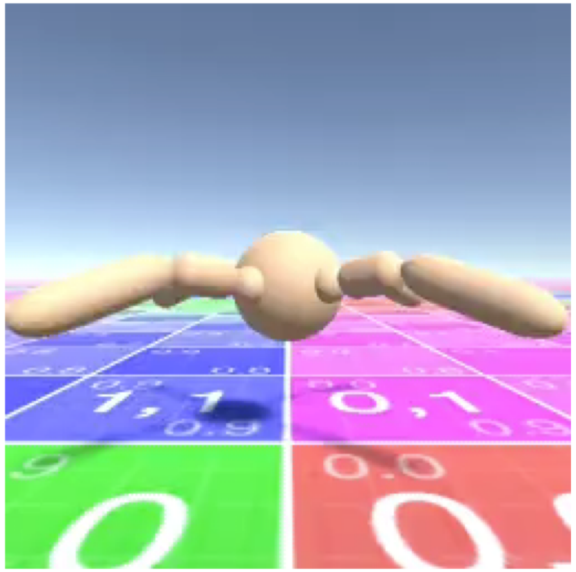
- ACKTOR (OpenAI Baselines)
  - Modified: Continuous Features / Continuous Features
- DDPG + Param noise (OpenAI Baselines)
  - Modified: fix bugs

Phases

1. Naïve attempts – ACKTOR
   - Simple Worm
   - MuJoCo Importer: OpenAI Ant
2. Structured - DDPG
   - Reward Function: OpenAI Gym, OpenAI Roboschool
   - MuJoCo Importer: OpenAI Hopper, Walker2d, Humanoid
3. More Structure
   - Refactored to use Customizable Joints (based on paper)
   - … Note: Broke Humanoid
   - Reward Function: DeepMind Paper
   - MuJoCo Importer: DeepMind Walker (add sensors)
4. Comparisons
   - Side by Side with OpenAI's MuJoCo

rl locomotion: results – Phase #1



Phase #1

- Naïve attempts – ACKTOR

Reward Function

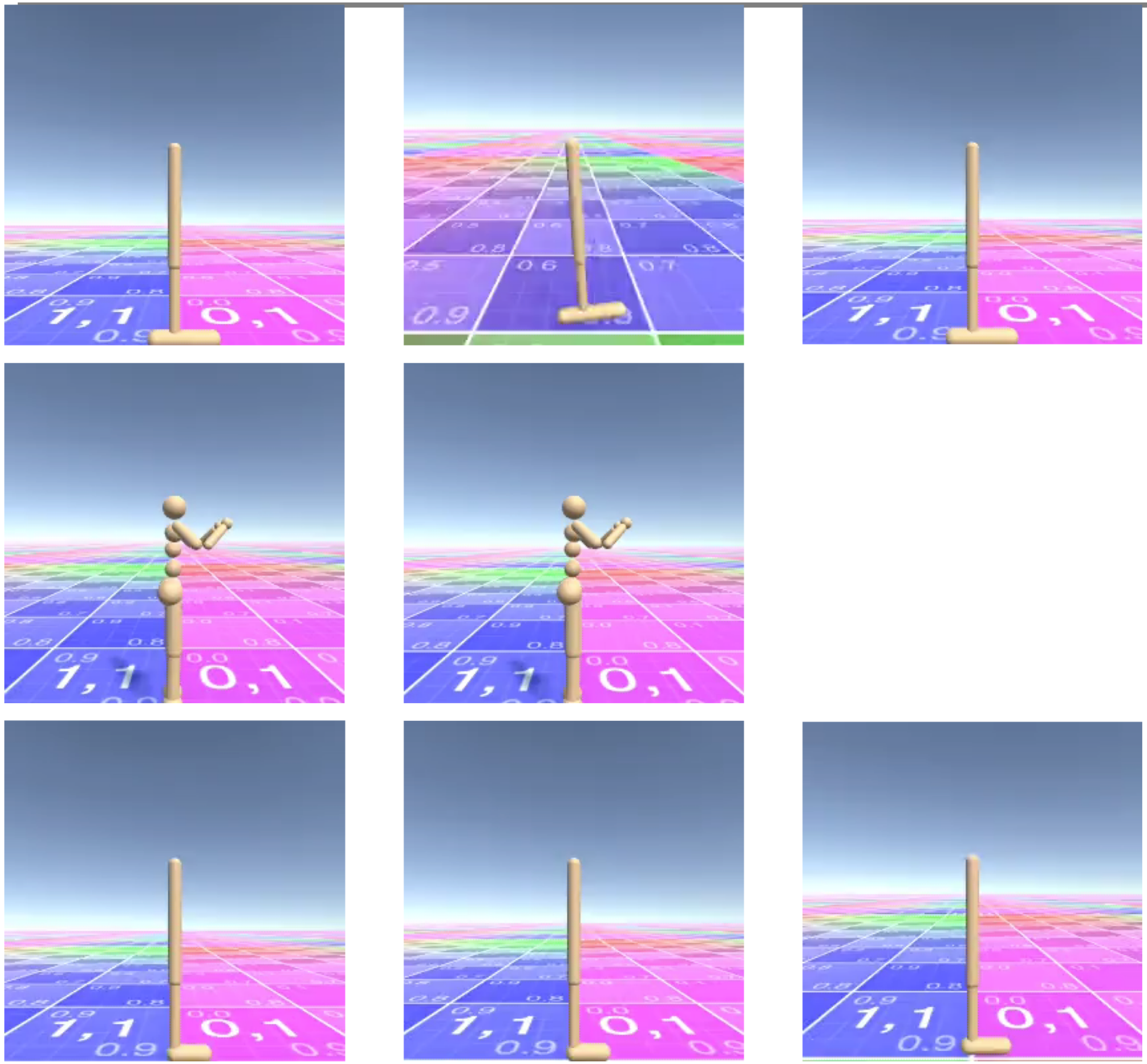
- Score = x velocity

No Termination Function

Scope - Ant

- 44 Experiments,
- 133 runs,
- 86,562,000 steps,
- ~90 clock hrs

rl locomotion: **results** – Phase #2

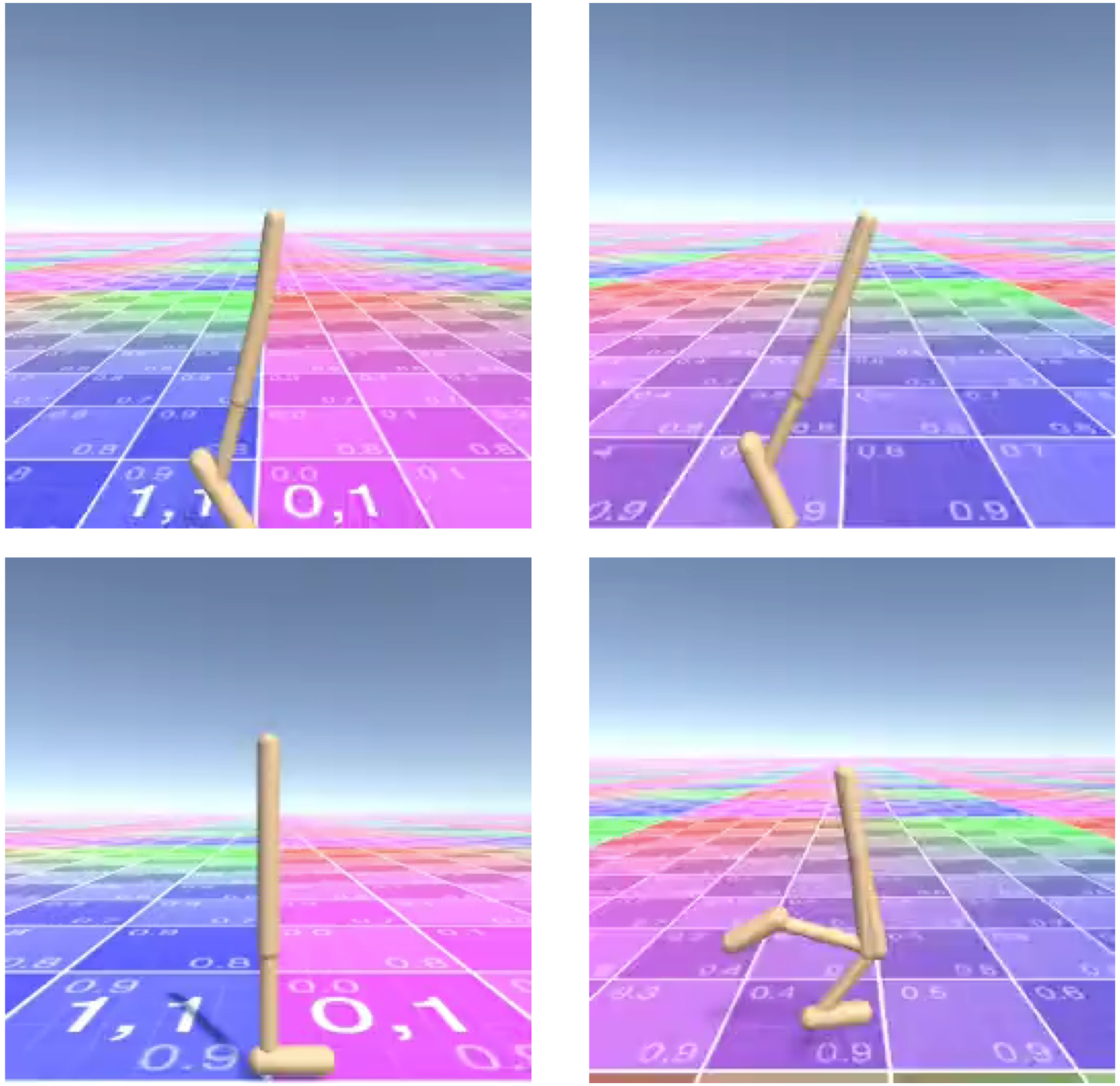


Phase #3

- Structured - DDPG

```
float StepReward_OaiHopper()
{
    var alive_bonus = 1f;
    var reward = _qvel[0];
    reward += alive_bonus;
    var effort = _actions
        .Select(x=>x*x)
        .Sum();
    reward -= (float) (1e-3 * effort);
    return reward;
}
```

rl locomotion: results – Phase #3



```
bool Terminate_HopperOai()
{
    if (Terminate_OnNonFootHitTerrain())
        return true;
    if (_qpos == null)
        return false;
    var height = _qpos[1];
    var angle = Mathf.Abs(_qpos[2]);
    bool endOnHeight = (height < .3f);
    bool endOnAngle = (angle > (1f/180f) * (5.7296f *6));
    return endOnHeight || endOnAngle;
}
```

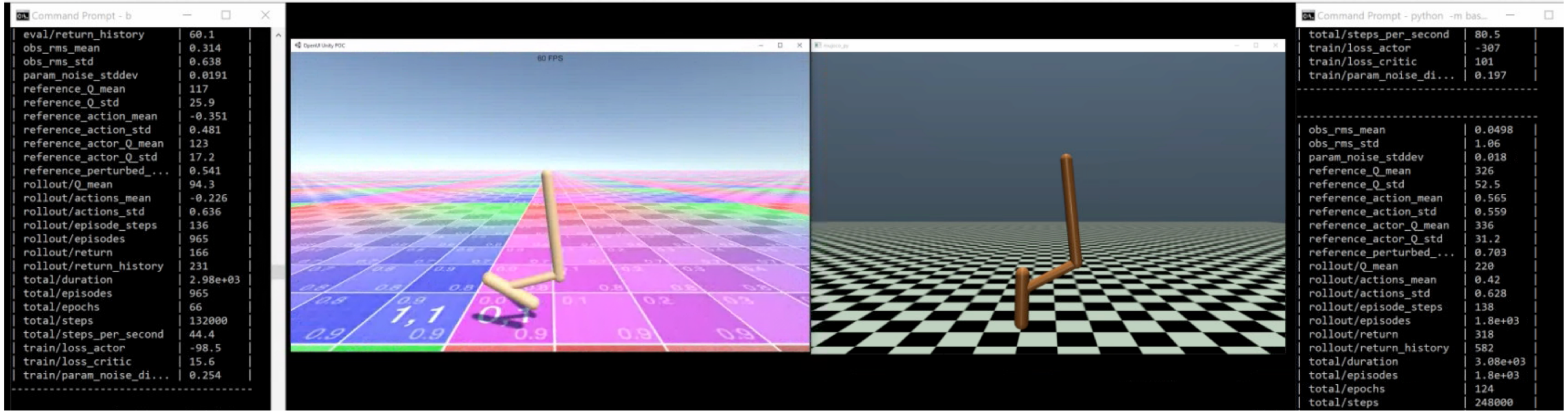
```
float StepReward_DmWalker()
{
    var feetYpos = _mujocoJoints
        .Where(x=>x.JointName.ToLowerInvariant().Contains("foot"))
        .Select(x=>x.Joint.transform.position.y)
        .OrderBy(x=>x)
        .ToList();
    var lowestFoot = feetYpos[0];
    var height = _qpos[1]-lowestFoot;

    var dt = Time.deltaTime;
    var rawVelocity = _qvel[0];
    var velocity = 10f * rawVelocity * dt;
    var uprightBonus = 0.5f * (2 - (Mathf.Abs(_qpos[2])*2)-1);
    var heightPenality = 1.2f - height;
    heightPenality = Mathf.Clamp(heightPenality, 0f, 1.2f);

    var reward = velocity+uprightBonus-heightPenality;

    return reward;
}
```

rl locomotion: results – Phase #4



Side by side comparison

- Training time ~5 %
- Training steps ~ 10%
- Style – different (i.e. they learned differently)

rl locomotion: conclusion

Outcomes

- Reproduced Hopper + Walker
- I Expect DeepMind results are reproducible
- Worthy of more study
- Expect MuJoCo to always have an edge

Future work:

- Implement humanoid
- Implement Unity models
- Implement learning from Mocap
- Mix learning from Mocap with reward
- Implement obstacles
- Explore HER, Meta Learning, Learning a Hierarchy

Total Scope

- 380 Experiments,
- 1,149 runs,
- 747,580,909 steps,
- ~780 clock hrs

publishing results

#2 learning an advance control in 99 training steps

learning non-trivial control in 99 training steps

Abstract:

- Many computer science problems have far less dimensionality than the robotics or playing video games from pixel domains that Reinforcement Learning research focuses on. However, small dimensional problems can still propose complexity that would benefit from RL. In this work we show that applying grid search to hyper parameters and using modern Reinforcement Learning algorithms can dramatically reduce the learning steps required to master a non-tribal task.

Environment:

- Modified Unity 'Tanks' tutorial where a 'dumb' tank drives towards and then explodes on impact with the 'rl' tank. The score function is 'if dumb destroyed then 100- number of shots fired.'
- The rl tank has two discrete actions (Null, Fire). The environment has 3 properties (DumbDistance: dumb tank to rl tank), (LaunchForce: number of game frames that fire has been held), DumbDistanceSimple: dumb tank distance in cells). Note: 'nstack' is a hyper-parameter to control the number of observation stacks (stacking previous n steps observations with current observation)
- In tanks, the fire power of the tank is determined by the length of time the user hold the fire button for. This gives us a non-trivial skill to learn.

Methods

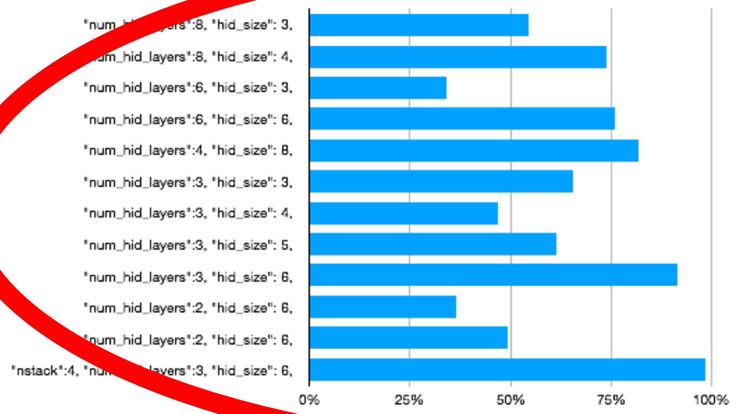
- Custom development environment (Analog) bridges OpenAI Algorithms and Unity. Tested modified versions of OpenAI's implementations of DQN and AKTOR (modified to work with Analog). Hyperparamerters number of layers, layer size, learning rate.
- Manual nested grid search was used applying Andrew Ng best practice approach (.001, .003, .01, .03, .1, .3, 1, 3, 10, 30, etc)
- Scoring is the mean of 100 runs. Each experiment consisted of 3-7 sub-experiments (meaning the score is the mean of 300-700 runs)

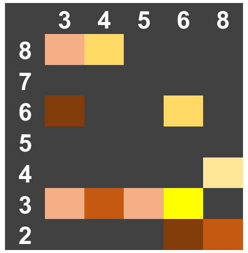
Train AI to hold fire to shoot at distance?

learning non-trivial control in 99 training steps

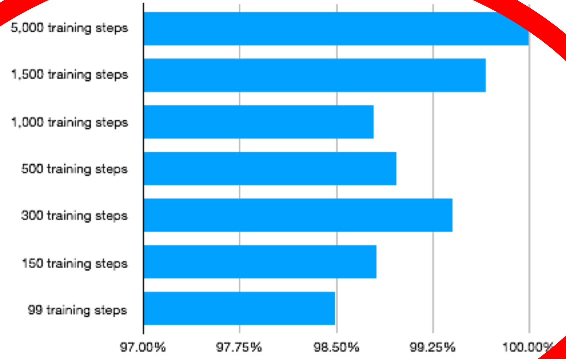
Results:

A grid search of 2-8 hidden layers of a size of 3-6 gives a wide range of results varying between a low of 34% and a high of 91% with standard deviation of .19. There is no decreeable pattern to why one set performance better than another, for example, the total number of connections does seams irrelevant as the the lowest score of 34% (6x3) and highest 91% (3x6) share the same number of connections. Increasing nstack to 4 (default is 3) improved the overall score to 98.49%. Note: further exploration of nstack was done with no impactful results.





Reducing the number of training steps progressively from 5,000 to 1,500, 1,000, 500, 300, 150, 99 showed no loss in the learning rate.



Conclusions

- We have shown that modern algorithms such as ACKTOR can learn non-trivial, low demential, discrete tasks in very few steps when an aggressive search strategy is used to find the optimal network design and hyper-paramaters. Further investigation into improvement over grid or random search would be welcome.

Grid Search

- Num layers * layer size
- * num stacks

Search

- Num training steps

publishing results

#3 delayed feedback

The case for real time server side RL

delayed feedback :

Abstract:

- Publish today with python backend?
- Inspiration: Human visual reaction delay ~250ms
- Inspiration: Multiplayer games

Method

- Train at 20fps with 10 step delay

Results

- ~30% deprecation in learning time

Conclusion

- Not silver bullet
- Worth more investigation

Thank you

joe@joebooth.com