

# **An Introduction to Reinforcement Learning**

by John Sigmon

# What is Reinforcement Learning (RL) ?

---

## Supervised Learning

A classification or regression problem where we have labeled data we can use to learn the model.

## Unsupervised Learning

A problem where there are no labels and the goal is to find hidden structure in the data.

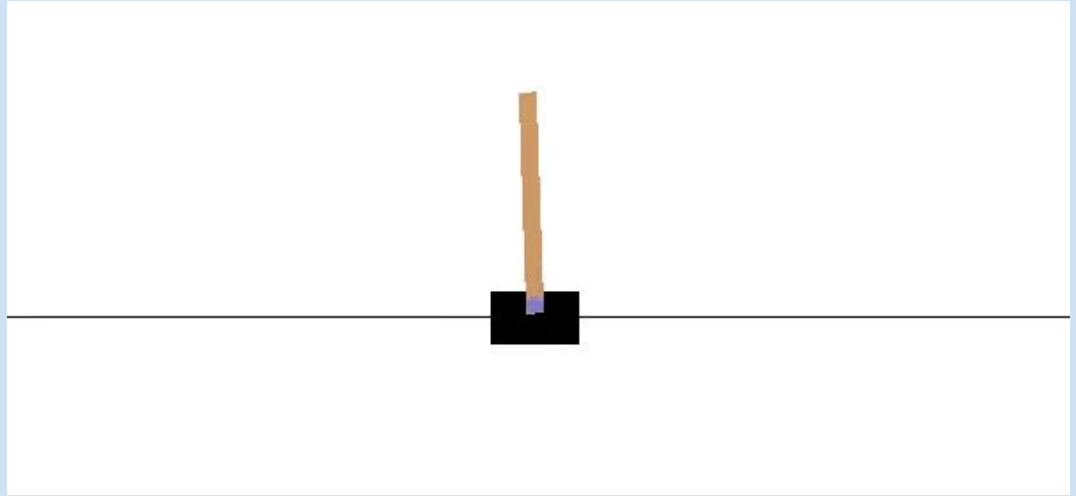
## Reinforcement Learning

A time dependent problem where the model must learn based on rewards.

# Our Go To Example

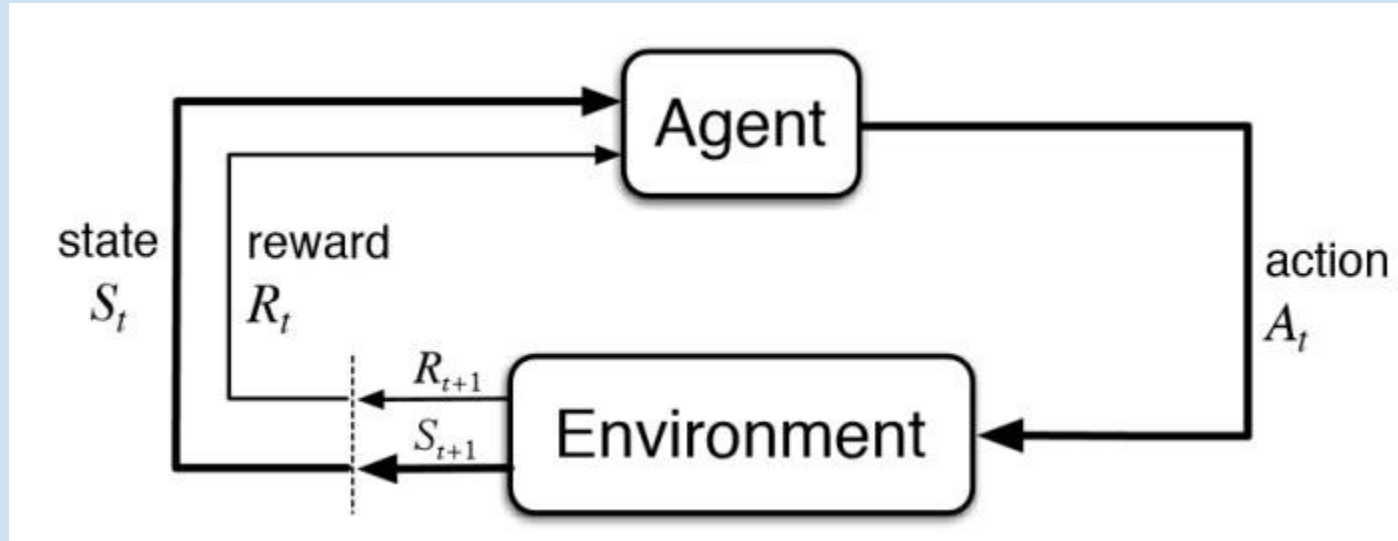
---

Cartpole



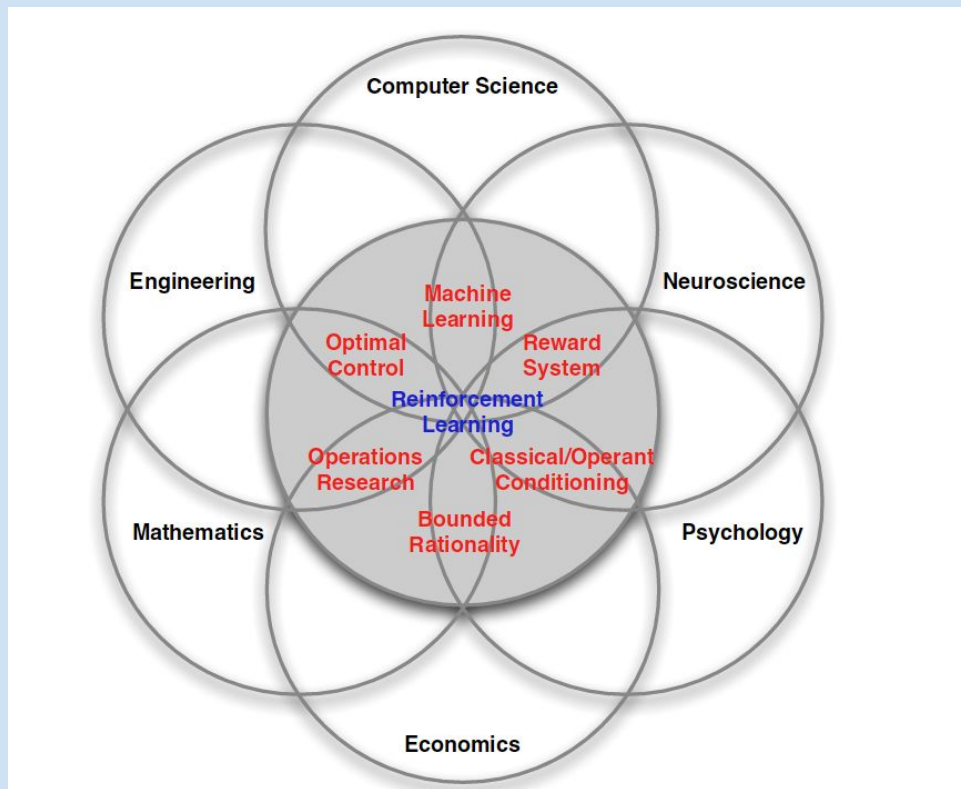
# What is Reinforcement Learning (RL) ?

Typically means solving any problem that fits into this diagram



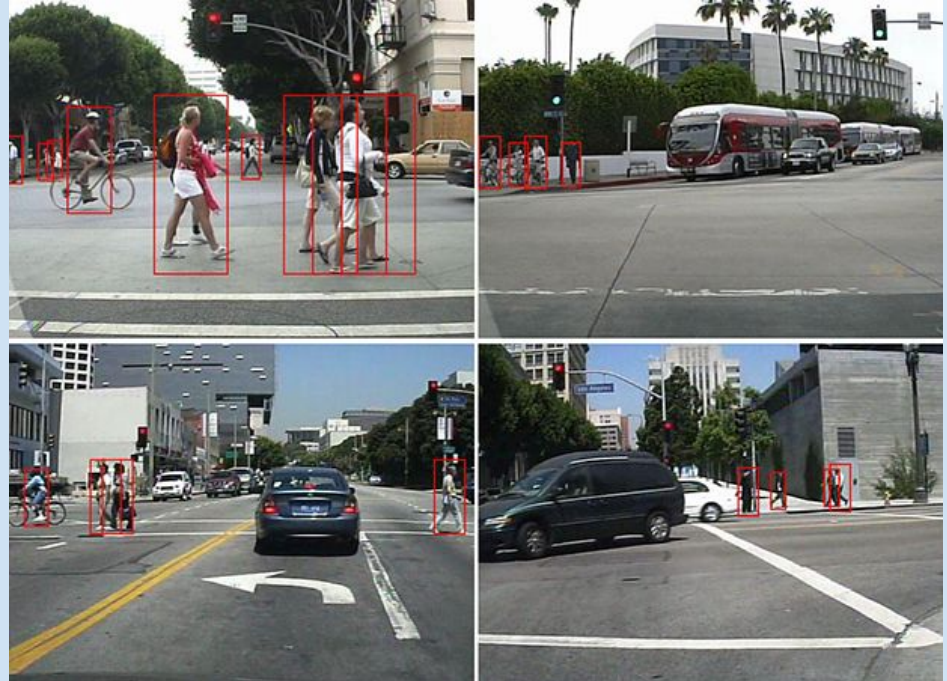
# What is Reinforcement Learning (RL) ?

How people (in RL) see the field



# Where is RL used?

## Self driving cars



<https://spectrum.ieee.org/cars-that-think/transportation/advanced-cars/deep-learning-makes-drive-less-cars-better-at-spotting-pedestrians>

# Where is RL used?

Playing video games



Space Invaders: <https://www.youtube.com/watch?v=UZIEXI9xgR8>

# Where is RL used?

Grasping objects in robotics





# Where is RL used?

Simulated robot soccer



# Where is RL used?

Chatbots (work in progress)



# The Details

First, difference in notation

RL Terminology	RL Notation	Control Terminology	Control Notation
Action	$a$	Control Signal	$u$
State	$s$ (or $o$ )	State / Environment	$x$
Reward	$r$	Cost / Payoff	$J$ or $c$

# The Details

---

## The Markov Decision Process:

$$\langle S, A, P_a, R_a, \gamma \rangle$$

<b>State</b>	Set of all possible states (usually infinite)	$S = \{s_1, s_2, \dots, s_n\}$
<b>Action</b>	Set of all possible actions (usually infinite)	$A = \{a_1, a_2, \dots, a_m\}$
<b>Dynamics</b>	Probability of a state given previous state and action	$Pr(s_{t+1}   s_t, a_t)$
<b>Reward</b>	Reward for transitioning to a state, given a previous state, action pair	$r(s_t, a_t, s_{t+1}) : S \times A \times S \rightarrow \mathbb{R}$
<b>Discounting Factor</b>	Discounting factor for future rewards	$\gamma \in [0, 1]$

# The Details

---

Policy:

A function from the state space to the action space.

$$\pi(s) : S \rightarrow A$$

Optimal Policy:

The policy that maximizes the sum of all future rewards.

$$\pi^* = \operatorname{argmax}_{\pi} \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t \cdot r_t(s_t, a_t, s_{t+1}) \mid \pi \right]$$

# Finding an Optimal Policy

---

Value Iteration

Policy Iteration

Dynamic Programming

Policy Gradients

Temporal Difference Learning

Sarsa Learning

Q Learning

Monte Carlo Methods

# How to Get Started

## OpenAI

Founded December 2015

Released OpenAI Gym April 2016

<https://gym.openai.com/>

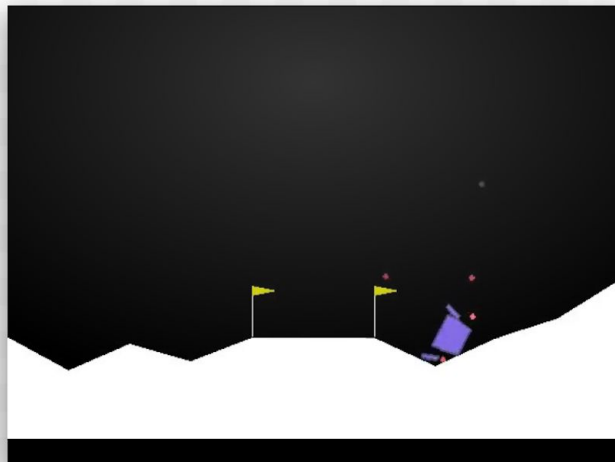


Gym

Gym is a toolkit for developing and comparing reinforcement learning algorithms. It supports teaching agents everything from walking to playing games like Pong or Pinball.

[View documentation >](#)

[View on GitHub >](#)



RandomAgent on LunarLander-v2

# Code Demo

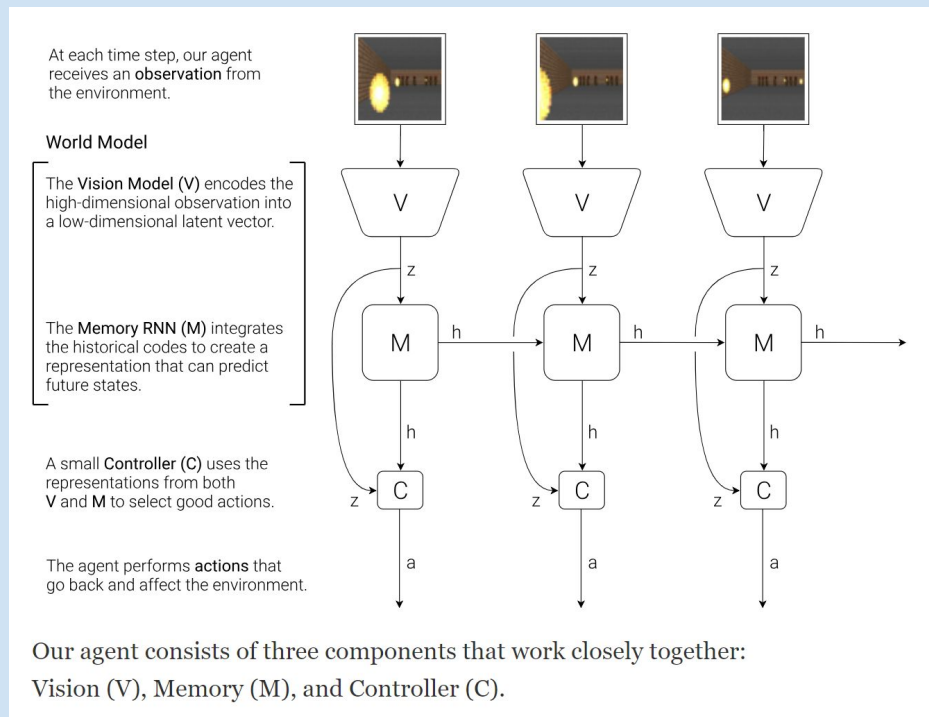
```
1  import gym
2  env = gym.make('CartPole-v0')
3  env.reset()
4  for _ in range(20):
5      observation = env.reset()
6      for t in range(100):
7          env.render()
8          print(observation)
9          action = env.action_space.sample()
10         observation, reward, done, info = env.step(action)
11         if done:
12             print("Episode finished after {} timesteps".format(t+1))
13         break
14
```



# Current Research

## World Models by David Ha, Jurgен Schmidhuber

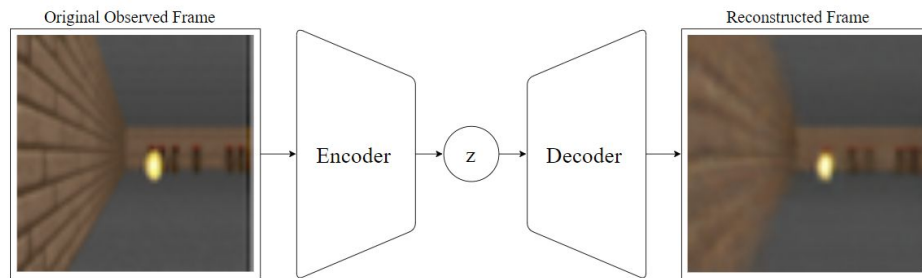
<https://arxiv.org/abs/1803.10122> and <https://worldmodels.github.io/>



# World Models

## Vision Model

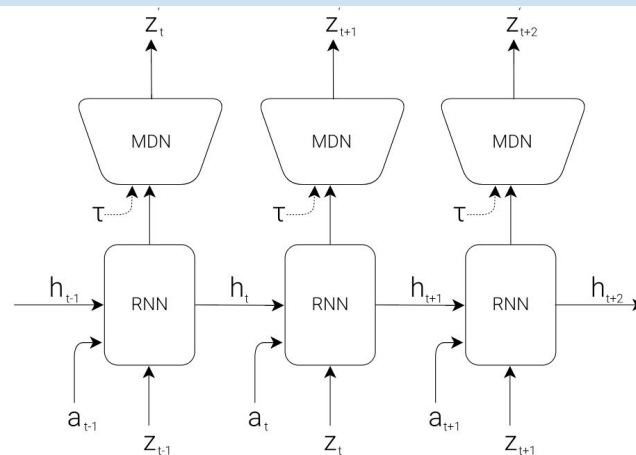
A variational autoencoder



Flow diagram of a Variational Autoencoder. [31, 32]

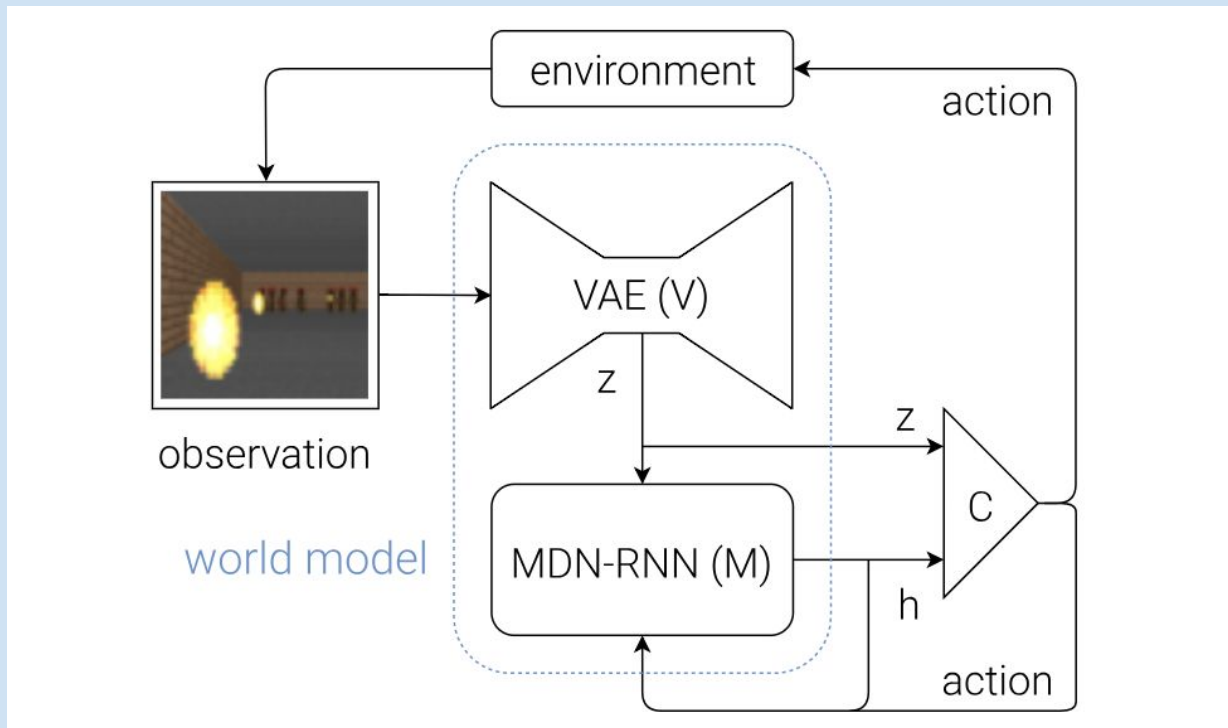
## Memory Model

RNN feeding into a Mixture Density Network



RNN with a Mixture Density Network output layer. The MDN outputs the parameters of a mixture of Gaussian distribution used to sample a prediction of the next latent vector  $z$ .

# World Models



# Current Research

---

World Models by David Ha, Jorgen Schmidhuber

<https://arxiv.org/abs/1803.10122> and <https://worldmodels.github.io/>

Deep Neuroevolution by Uber AI Labs

<https://arxiv.org/abs/1712.06567> and <https://eng.uber.com/deep-neuroevolution/>

Learning Dexterous In-Hand Manipulation by Open AI

<https://arxiv.org/abs/1808.00177> and <https://blog.openai.com/learning-dexterity/>

One Shot Imitation Learning by Open AI

<https://arxiv.org/abs/1703.07326> and <https://blog.openai.com/robots-that-learn/>

Reinforcement Learning with Deep Energy Based Policies

<https://arxiv.org/abs/1702.08165>

# Educational Resources

---

*Reinforcement Learning* by Richard Sutton, Andrew Barto

available online: <http://incompleteideas.net/book/bookdraft2017nov5.pdf>

Richard Sutton's home page

<http://incompleteideas.net/>

Peter Stone's course on Reinforcement Learning

<http://www.cs.utexas.edu/~pstone/Courses/394Rfall16/resources/index.html>

Levin, Schulman, and Finn's course on Deep Reinforcement Learning

<http://rll.berkeley.edu/deeprlcoursesp17/>

David Silver's course on Reinforcement Learning

<http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html>

# Some Other RL Environments

---

VizDoom for playing the 1993 video game Doom

<http://vizdoom.cs.put.edu.pl/>

TorchCraft a library to connect Torch and RTS games like StarCraft

<https://github.com/TorchCraft/TorchCraft>

DeepMind Lab a 3d environment based on Quake III

<https://github.com/deepmind/lab>

Dopamine a Google RL framework

<https://ai.googleblog.com/2018/08/introducing-new-framework-for-flexible.html>

<https://github.com/google/dopamine>

# Thank You

---



Twitter

[@JohnSigmon](https://twitter.com/JohnSigmon)



LinkedIn

<https://www.linkedin.com/in/john-sigmon/>



Email

[johnsigmon@gmail.com](mailto:johnsigmon@gmail.com)



GitHub Page for this talk

[https://github.com/jsigee87/rl\\_talk](https://github.com/jsigee87/rl_talk)