


1 Continuous Mountain Car

The continuous mountain car task consists in driving an underpowered (gravity is stronger than the engine) car up a mountain and out of a valley. The task is successfully completed once the point on the horizontal axis that matches the top of the mountain is reached. The simplified model of the environment, ~~presented in its discrete version in [1]~~, will be implemented. This is also the implementation presented in OpenAI's *Gym* Python library [2]: 

Observed variables:

1. $p \in [p_{min} = -1.2; p_{max} = 0.6]$: position of the car on the horizontal axis.
2. $v \in [v_{min} = -0.07; v_{max} = 0.07]$: component of the car's velocity that is parallel to the horizontal axis.

Action:

1. $f \in [-f_{max}; f_{max}]$: can be interpreted as the car's power, thrust, or acceleration. It has three main states: forward thrust ($f > 0$), no thrust ($f = 0$), and reverse thrust ($f < 0$). Forward thrust is considered towards the right of the track.

The shape of the track is described by $h = \sin(C_h p)$. The initial position is selected randomly (sampling from a uniform distribution) in the vicinity of the the bottom of the valley (where $C_h p \approx -\frac{\pi}{2}$). The initial velocity of the car is always zero.

State space equations:

$$\begin{aligned} v_{t+1} &= v_t + C_f f_t - C_g \cos(C_h p_t) \\ p_{t+1} &= p_t + v_{t+1} \end{aligned} \tag{1}$$

where C_f is a power coefficient that describes how the action influences the change in velocity, $C_g = 0.0025$ summarizes the effects of the force of gravity, and $C_h = 3$ describes the shape of the track.

The reward for the mountain car environment is defined as -1 for every time step in which the top of the mountain isn't reached; which is achieved when $p_{t+1} = p_{max}$ for the first time. When $p_{t+1} = p_{min}$, the cars velocity is reset to 0.

1.1 Operating Modes

Table 1 summarizes the values of the parameters that define the different operating modes for the mountain car system.

Table 1: Operating modes for the Mountain Car system

Parameter	Typical Values	Lower Bound	Upper Bound
C_f	0.001	$C_{f_{min}}$	$C_{f_{max}}$

1.1.1 Parameter C_f

Variations in the power coefficient for fixed action bounds $[-f_{max}; f_{max}]$ result in changes to power transfer. A reduction of this coefficient could indicate engine faults, wear in the car’s bearings, inadequate lubrication, among others.

Maximum C_f is defined as $C_{f_{max}} = \frac{C_g}{f_{max}} \cos(C_h p_{max})$, ~~which is~~ the value for which the car’s engine, at full throttle, is stronger than the force of gravity for every point of the track, rendering the task trivial.

Minimum C_f , on the other hand, should not be too small that it makes it impossible for the car to climb the mountain at full throttle with maximum built up velocity. It’s exact value is not as trivial, this is as far as I’ve narrowed it down:

$$C_{f_{min}} = \frac{2}{n(n+1)f_{max}}(p_{max} - p_{min} + C_g \sum_{i=1}^n \text{icos}(C_h p_{t+n-i})) \quad (2)$$

where n is the number of time steps it takes the car to get from p_{min} to p_{max} at full throttle.

The boundary value $C_{f_{min}}$ can also be determined empirically by driving the car from p_{min} to p_{max} at full throttle, and selecting the smallest value for which the car reaches the top of the mountain.

1.1.2 Parameter C_h

C_h determines the shape of the track, regulating the steepness of the mountain. A higher value of C_h results in a steeper track. The selection of C_h is directly related to the boundaries of the position state variable, since $C_h p_{max} \approx \frac{\pi}{2}$ and $-\frac{3\pi}{2} < C_h p_{min} < -\frac{\pi}{2}$.

However, the relationship between C_h and the coefficients C_g and C_f is unclear (at least for every paper I’ve checked), since the values $C_g = 0.0025$, $C_f = 0.001$, and $C_h = 3$ are always selected arbitrarily; therefore, I cannot properly model the variations in the steepness of the track.

2 Cart Pole

The inverted pendulum on a cart, or cart-pole problem, consists in balancing an unstable pole connected to a cart, by moving the cart (and the pole’s pivot point) horizontally. The task is failed if the pole falls under a certain angle or the

cart moves out of its horizontal boundaries. The non-friction model presented in [3] will be implemented. This is also the implementation presented in OpenAI’s *Gym* Python library [2]:

Observed variables:

1. $x \in [x_{min}; x_{max}]$: position of the cart on the horizontal axis.
2. $\dot{x} \in \mathbb{R}$: velocity of the cart on the horizontal axis.
3. $\theta \in [\theta_{min}; \theta_{max}]$: angle of the pole.
4. $\dot{\theta} \in \mathbb{R}$: angular velocity of the pole.

Action:

1. $f \in [-f_{max}; f_{max}]$: the force applied to the cart to move it. A positive value is considered to be pushing right.

All initial observation variables are assigned a uniform random value in $[-0.05; 0.05]$.

State space equations:

$$\begin{aligned}
\ddot{\theta}_t &= \frac{g \sin \theta_t + \cos \theta_t \left(\frac{-f_t - m_p l \dot{\theta}_t^2 \sin \theta_t}{m_c + m_p} \right)}{l \left(\frac{4}{3} - \frac{m_p \cos^2 \theta_t}{m_c + m_p} \right)} \\
\ddot{x}_t &= \frac{f_t + m_p l (\dot{\theta}_t^2 \sin \theta_t - \ddot{\theta}_t \cos \theta_t)}{m_c + m_p} \\
\dot{x}_{t+1} &= \dot{x}_t + \ddot{x}_t \Delta t \\
x_{t+1} &= x_t + \dot{x}_t \Delta t \\
\dot{\theta}_{t+1} &= \dot{\theta}_t + \ddot{\theta}_t \Delta t \\
\theta_{t+1} &= \theta_t + \dot{\theta}_t \Delta t
\end{aligned} \tag{3}$$

where $g = 9.8$ is the acceleration of gravity; m_p and m_c are the mass of the pole and the cart respectively; $2l$ is the length of the pole; and Δt is the sampling period, or the time between two time steps. The state variables $\ddot{\theta}$ and \ddot{x} are the angular acceleration of the pole, and the acceleration of the cart, respectively, which are not **observable**.

The reward of the cart-pole task is defined as +1 for every time step, including the failure step.

2.1 Operating Modes

Table 2 summarizes the values of the parameters that define the different operating modes for the cart-pole system.

Table 2: Operating modes for the cart-pole system

Parameter	Typical Values	Lower Bound	Upper Bound
μ_c	$\{0.0005, 0.01, 0.05\}$	0	0.8
μ_p	$\{0.000002, 0.001\}$	0	0.01

2.1.1 Adding friction

The previous model can be modified to consider friction between the cart and the track (μ_c) and friction in the joint between the pole and the cart (μ_p). The state equations for $\ddot{\theta}$ and \ddot{x} are modified as follows:

$$\begin{aligned}
 N_{ct} &= (m_c + m_p)g - m_pl(\ddot{\theta}_t \sin\theta_t + \dot{\theta}_t^2 \cos\theta_t) \\
 \ddot{\theta}_t &= \frac{g \sin\theta_t + \cos\theta_t \left\{ \frac{-f_t - m_p l \dot{\theta}_t^2 [\sin\theta_t + \mu_c \operatorname{sgn}(N_{ct} \dot{x}) \cos\theta_t]}{m_c + m_p} + \mu_c g \operatorname{sgn}(N_{ct} \dot{x}) \right\} - \frac{\mu_p \dot{\theta}_t}{m_p l}}{l \left\{ \frac{4}{3} - \frac{m_p \cos\theta_t}{m_c + m_p} [\cos\theta_t - \mu_c \operatorname{sgn}(N_{ct} \dot{x})] \right\}} \\
 \ddot{x}_t &= \frac{f_t + m_pl(\dot{\theta}_t^2 \sin\theta_t - \ddot{\theta}_t \cos\theta_t) - \mu_c N_{ct} \operatorname{sgn}(N_{ct} \dot{x})}{m_c + m_p}
 \end{aligned} \tag{4}$$

where $\mu_p \dot{\theta}$ is a torque associated with the friction in the joint between the cart and the pole. N_c is the magnitude of the force normal to the track, and, for common choice of parameters, it will always be positive, as the cart should not try to jump off the track.

Minimum values for μ_p and μ_c are 0, for the cases with no friction. The maximum value for μ_c is selected as 0.8, following the worst case scenario in [4]. The maximum value for μ_p is selected as 0.01 according to its typical values.

References

- [1] Singh S, Sutton R. Reinforcement Learning with Replacing Eligibility Traces. Machine Learning. 1996;(22):123-58.
- [2] Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J, et al. OpenAI Gym; 2016.
- [3] Florian V. Correct equations for the dynamics of the cart-pole system. 2007. Available from: https://coneural.org/florian/papers/05_cart_pole.pdf.
- [4] Manrique Escobar CA, Pappalardo CM, Guida D. A Parametric Study of a Deep Reinforcement Learning Control System Applied to the Swing-Up Problem of the Cart-Pole. Applied Sciences. 2020;(10):9013.