Mapa 1a

**Block -> Linear; Resource -> Linear**

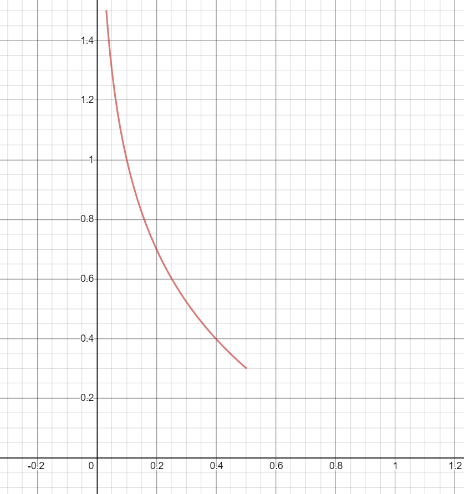
Com ambas as funções de ativação lineares o agente, se não se limitar as forças aplicadas ao agente, este irá sempre acabar por bater nas paredes. Ao apanhar o último recurso, este aproxima-se com velocidade muito elevada devido à distância entre o último e o antepenúltimo. O esforço de limite da força tem de ser maior para que não bata nas paredes, por isso é que o tempo é superior.

**Block -> Linear; Resource -> Gaussiana**

Relativamente ao anterior, este demonstrou uma aproximação menos abrupta ao recurso, o que é positivo.

**Block -> Linear; Resource -> Log neg**

A logarítmica negativa mostrou-se a melhor função de ativação para recursos, quando a função de ativação dos blocos é linear. Uma aproximação, em decrescendo, até ao recurso, o que é o melhor neste mapa, de forma a não bater nas paredes.

**Block -> Log; Resource -> Linear**

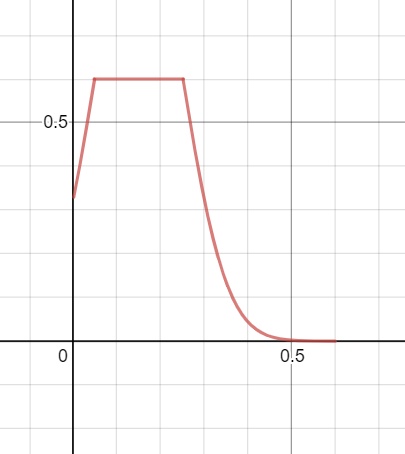
Block

Muitas vezes acontece que f(x) > 1, logo se meter o limite superior de y = 1, o resultado vai ser muitas vezes 1, então aumentei o limite superior para y = 1.5. Dimui o limite superior de x, porque a força quase nunca passa de 0.5.

**Block -> Log; Resource -> Gaussiana**

Block

Igual a anterior

Resource

Strength costuma estar perto de 0.1/0.2 sem variar muito, logo escolhi média = 0.15 e variância 0.1. O limite de x a variar entre 0 e 0.6, pois a partir de x > 0.6 f(x) = 0. Limites y de 0 a 0.6, para limitar que a força seja muito grande e possivelmente bata nas paredes.

**Block -> Log; Resource -> Log**

Block

Igual

Resource

Igual ao block

Pouco a fazer, mapa pequeno e fácil. Acontece que como o mapa é pequeno e com poucos obstáculos fica difícil ver muitas diferenças em cada função. Como os pickups estão muito próximos, não existe muito tempo para ver grandes variações, principalmente na função logarítmica, pois o robô ganha velocidade quando está longe, mas depois demora algum tempo a perder.

Mapa 1b

**Block -> Log; Resource -> Linear**

Completa o mapa todo, wR = 1, wB = 0.2. Resource com limites x e y de 0 a 1. Block com limites de y de 0 – 1.5 e x de 0 – 1. Tempo de 12s. Por vezes bate na parede depois de apanhar um recurso que está perto da parede.

**Block -> Log; Resource -> Gaussiana**

Completa o mapa todo, wR = 1, wB = 0.5. Resource com media 0.2 e variância de 0.06, limites x de 0 a 1 e y de 0.1 – 0.6, pois se meter 0 por vezes está muito perto do recurso e não apanha porque o valor é muito baixo e prefere fugir às paredes e 0.6 porque mais que isso é uma velocidade excessiva para o mapa em questão. Block com limites de y de 0 – 1.5 e x de 0 – 1. Tempo de 16s. Por vezes bate nos cantos das paredes.

**Block -> Log; Resource -> Gaussiana**

Completa o mapa todo, wR = 0.2, wB = 0.2. Resource e Block com limites de y de 0 – 1.5 e x de 0 – 1. Tempo de 16s. Por vezes bate nos cantos das paredes. Tempo de 11s. Não bate nas paredes. Parece ser a melhor opção, o robô muito cauteloso e ainda assim rápido quando está longe de paredes e recursos.

Mapa 2a

**Resource -> Log**

Poderá até existir alguma configuração com valores muito pequenos tornem possível o robô completar o mapa, no entanto com um comportamento linear o robô chega ao recurso com o máximo de velocidade possível, o que torna difícil que consiga para por completo antes de cair para o abismo. Agora caso, limitemos que a velocidade máxima seja muito pequena aí consegue parar.

**Resource -> Gaussiana**

Completa o mapa sem cair. wR =0.1. Limites de y de 0 – 0.6, média de 0.15 e variância de 0.03.  
Média adequa-se aos valores da força linear sobre o recurso, sendo que a variância foi determinada com base nos testes. O robô com variância > 0-03 chegava ao recurso com velocidade a mais.

**Resource -> Log**

Completa o mapa sem cair. wR = 0.01. Limites de y de 0 – 3, pois por vezes -log(x) > 1. Logo se tivéssemos limite superior de y = 1, estaríamos a simular uma função constante por alguns momentos.