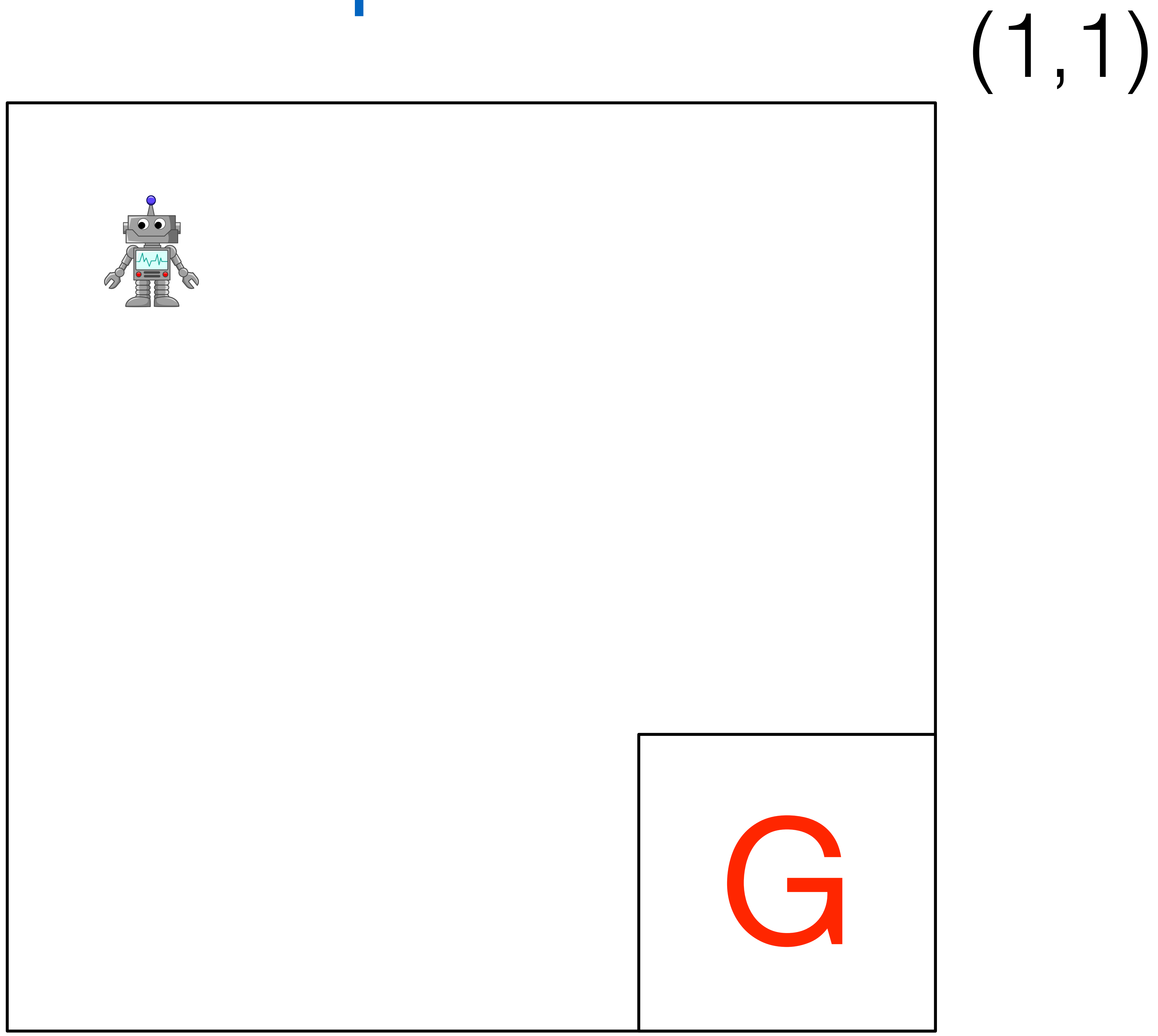
Course 3, Module 1
On-policy Prediction with Approximation

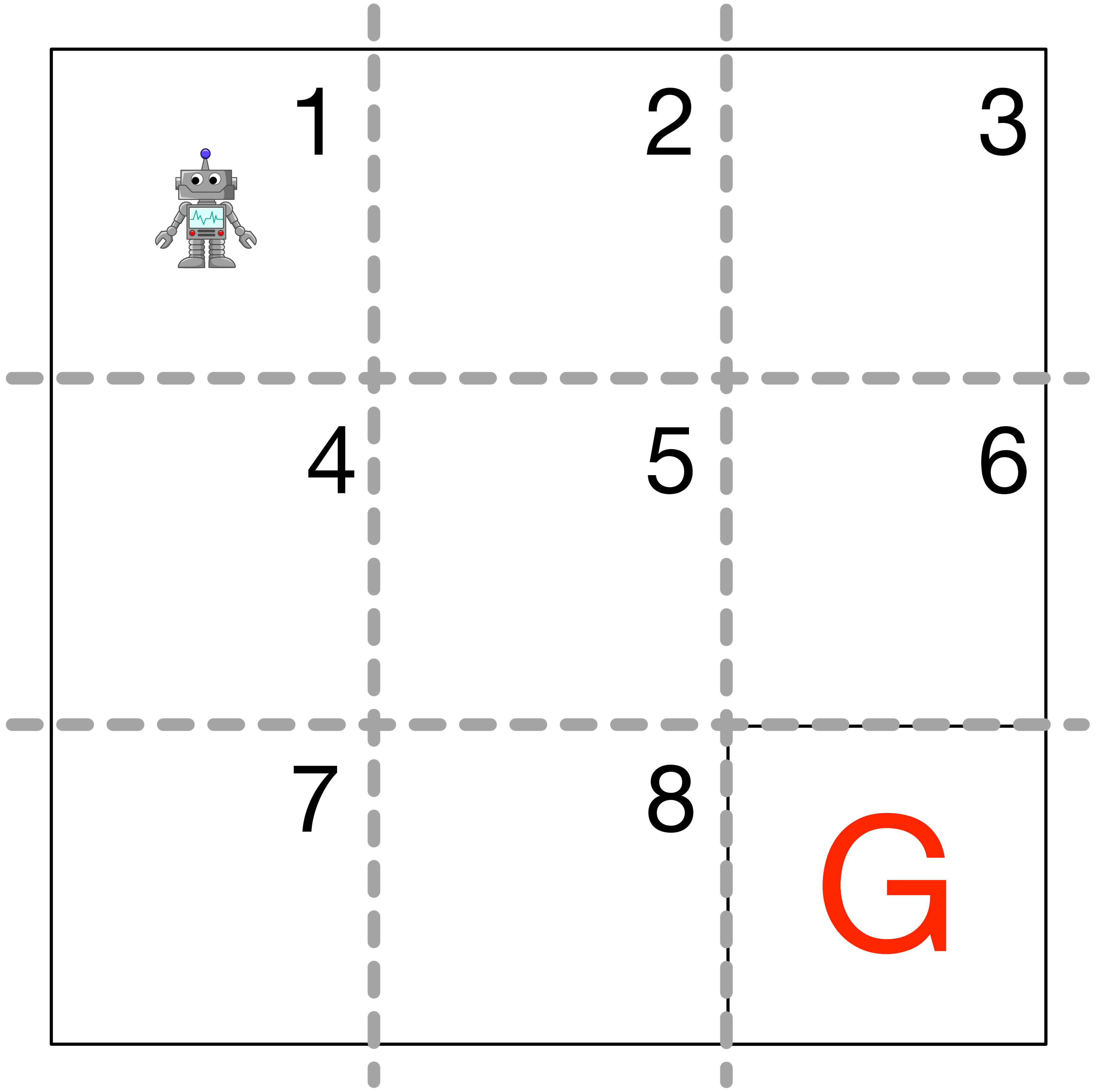
CMPUT 397
Fall 2019

- Link for questions:
  - **http://www.tricider.com/brainstorming/3D4V06mUv2V**

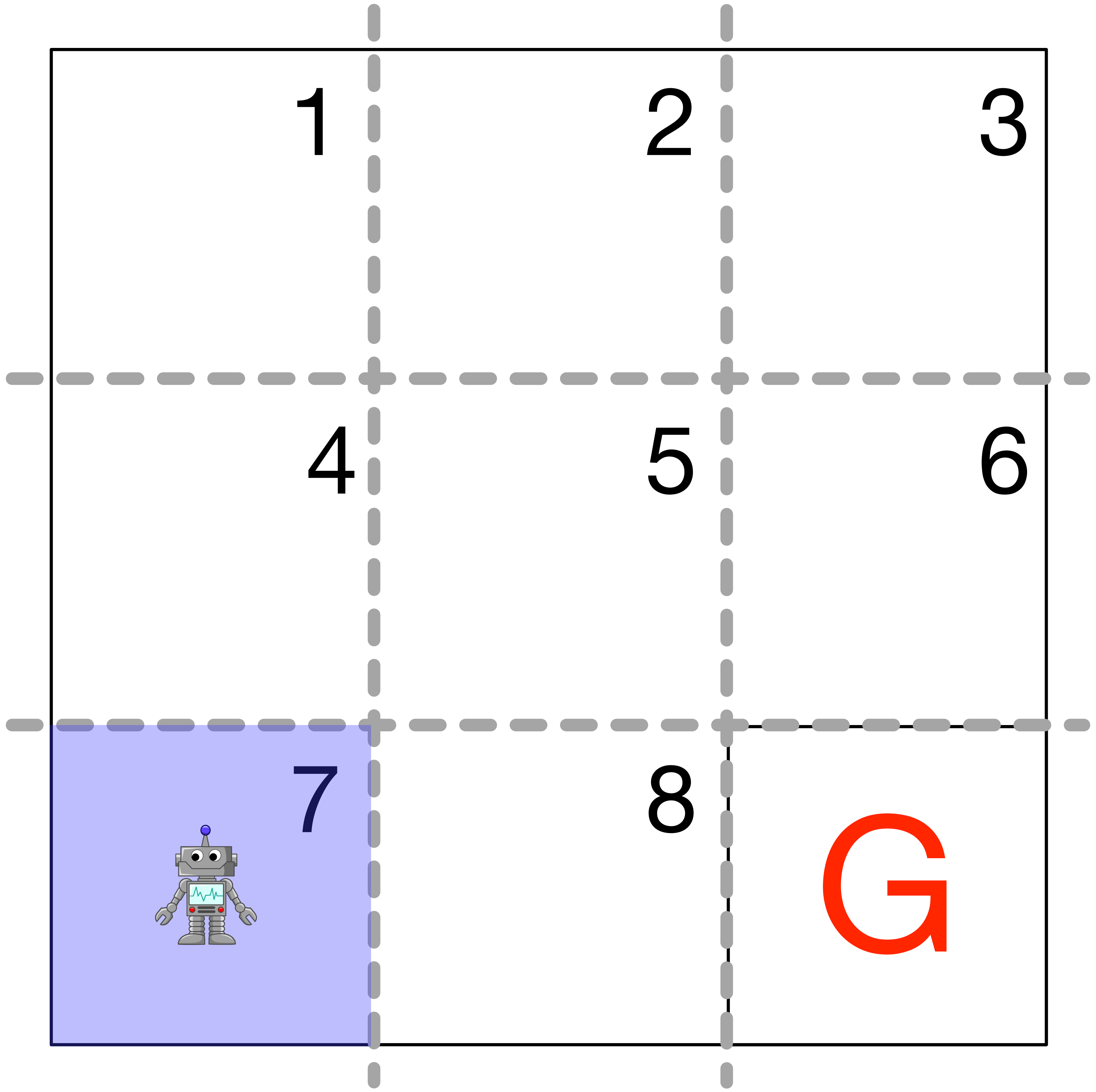
Imagine a continuous state space



Let's look at a simple state aggregation

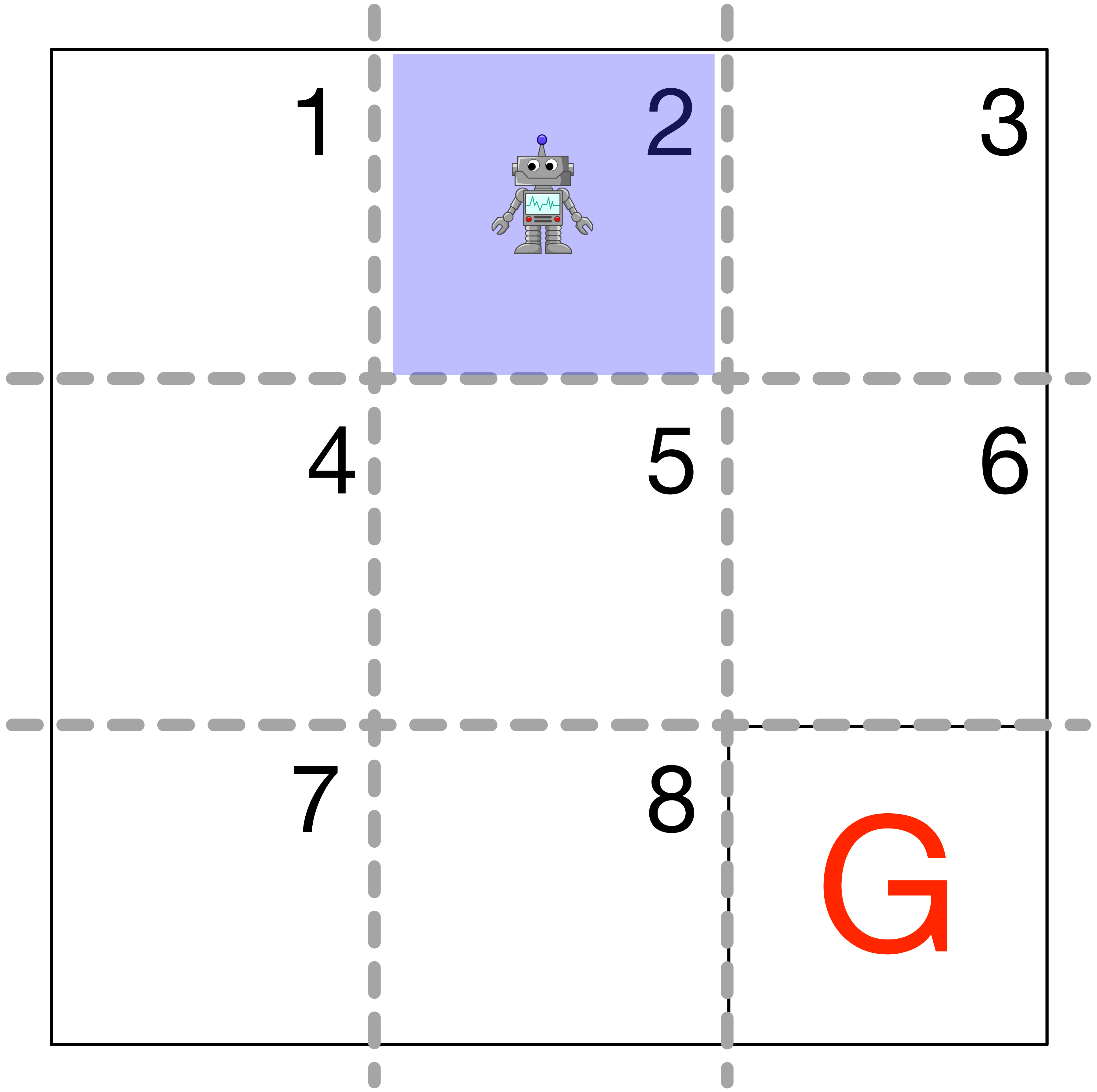


What would the feature vector be if the agent was somewhere in the bottom left?



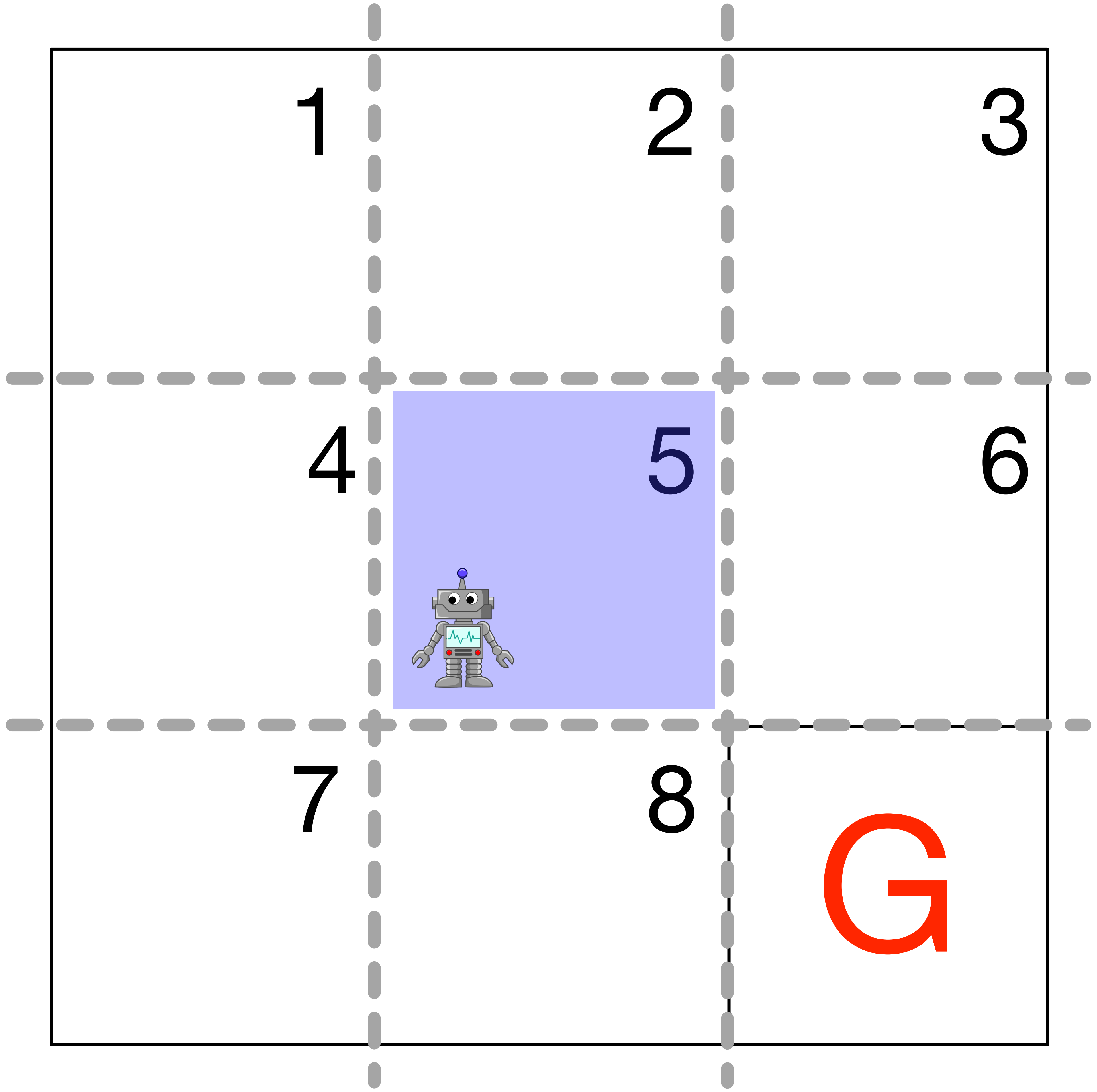
$$\boldsymbol{x}(s) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

What would the feature vector be if the agent was somewhere in the top middle?



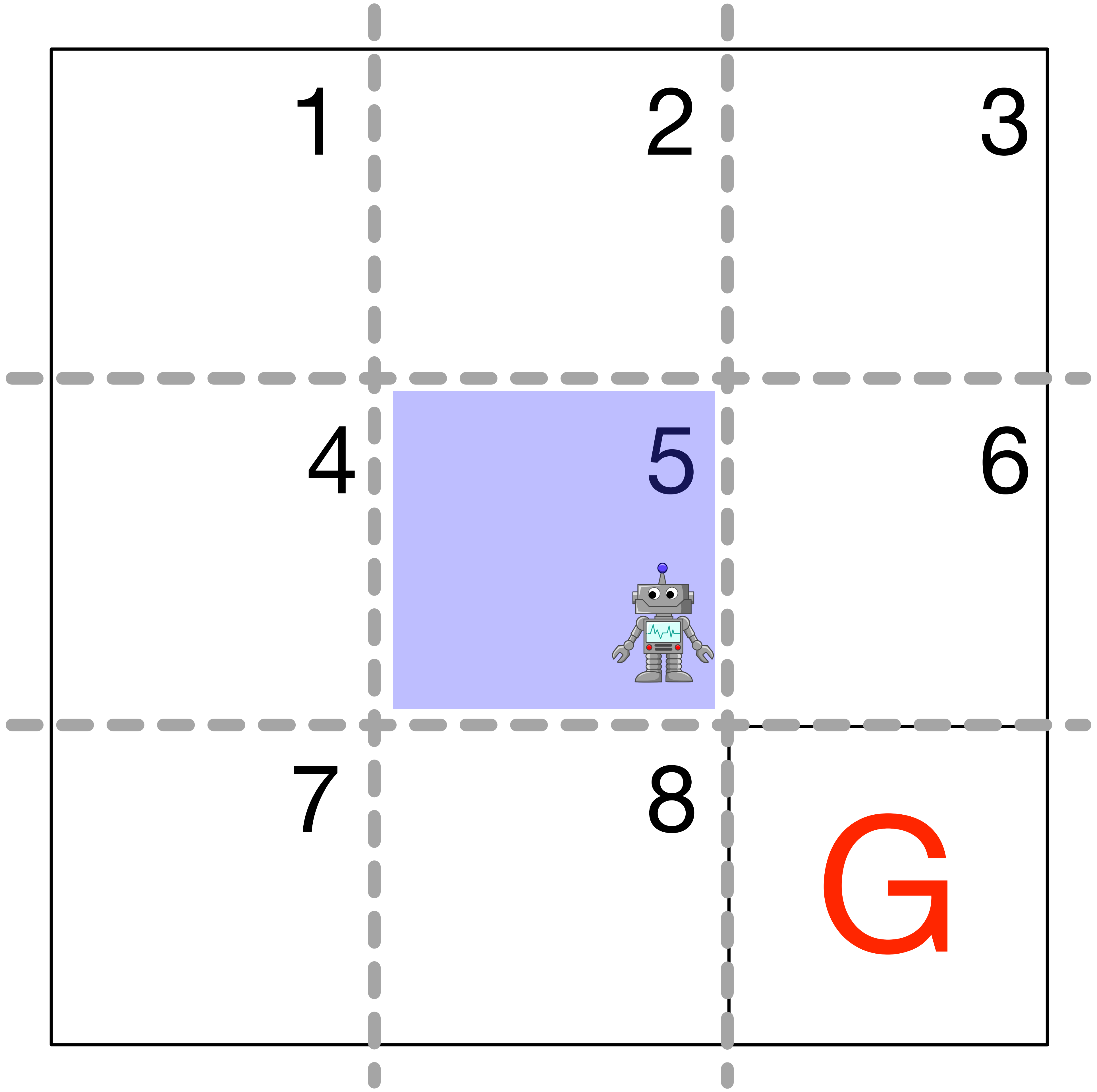
$$\boldsymbol{x}(s) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

What would the feature vector be if the agent was in the middle?



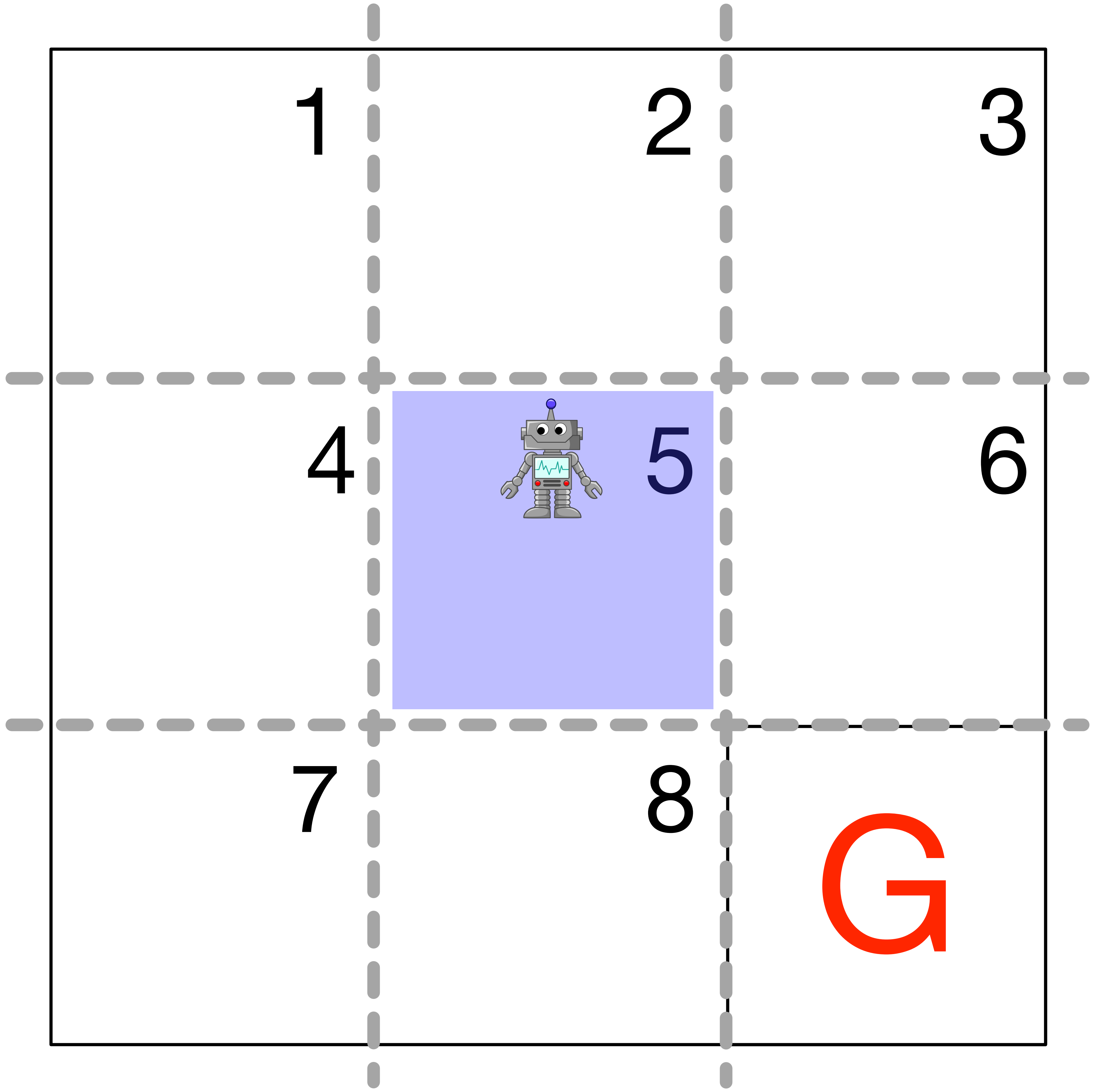
$$\boldsymbol{x}(s) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

How about here?



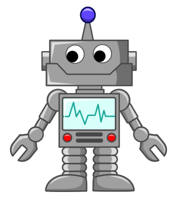
$$\boldsymbol{x}(s) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

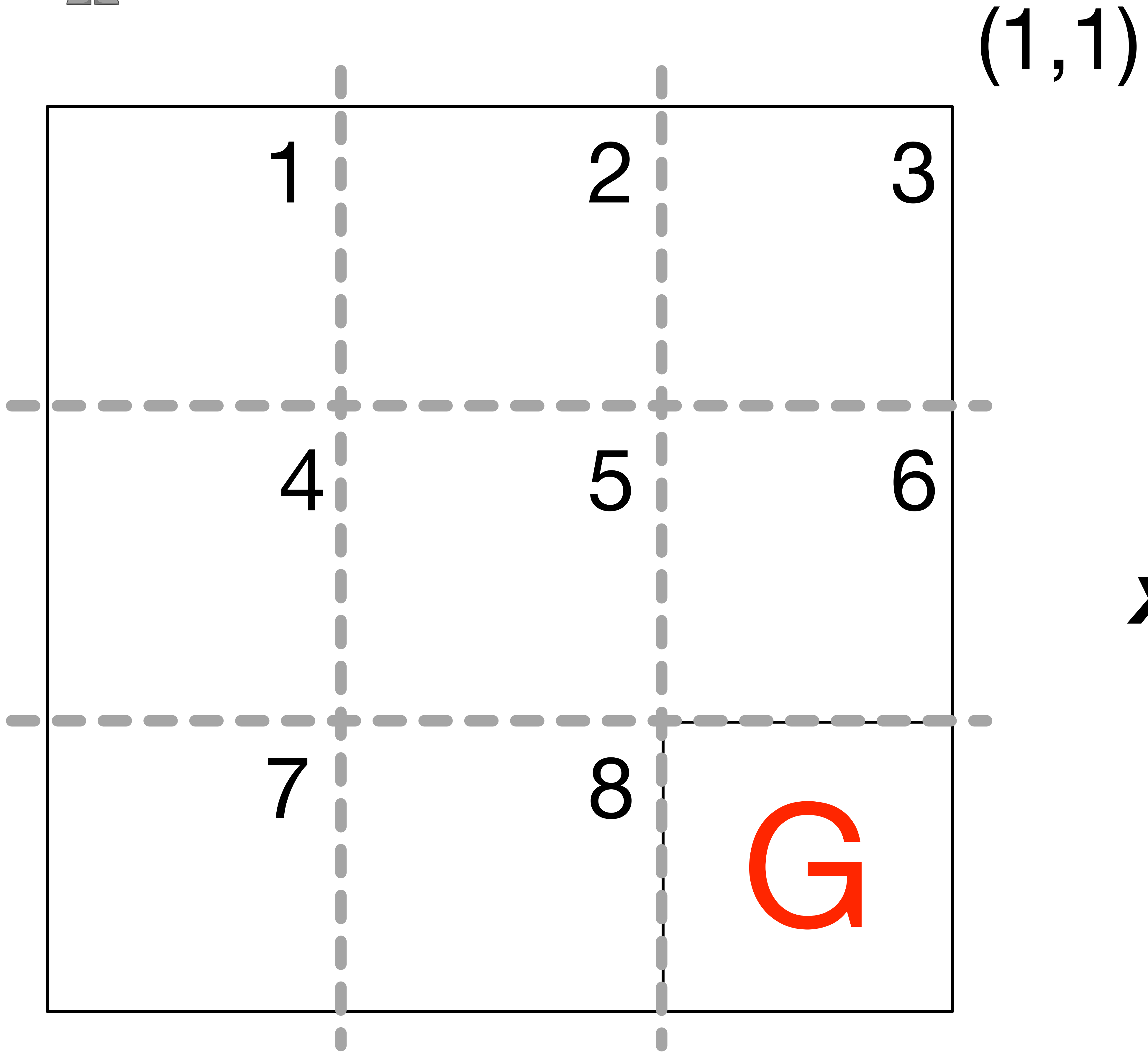
Or here?

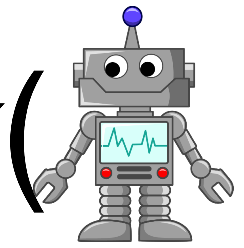


$$\boldsymbol{x}(s) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

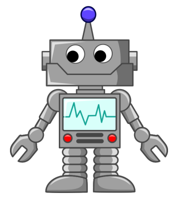
What would the feature vector be if the agent was at this point?

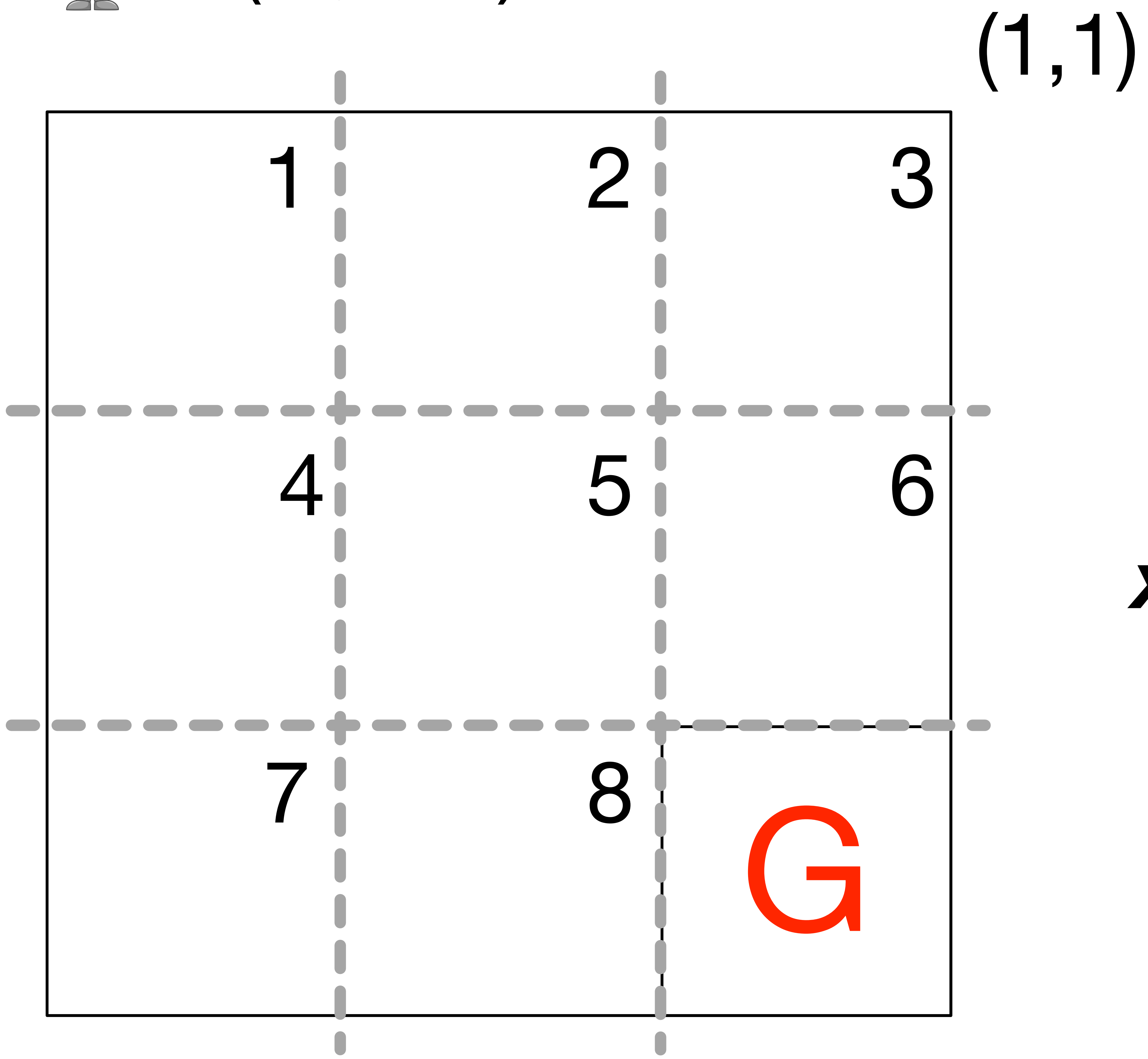
 = (.8, .8)

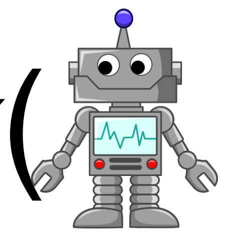


$\boldsymbol{x}(\mathrm{s}) =$ $\boldsymbol{x}($$) = \boldsymbol{x}(0.8, 0.8) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

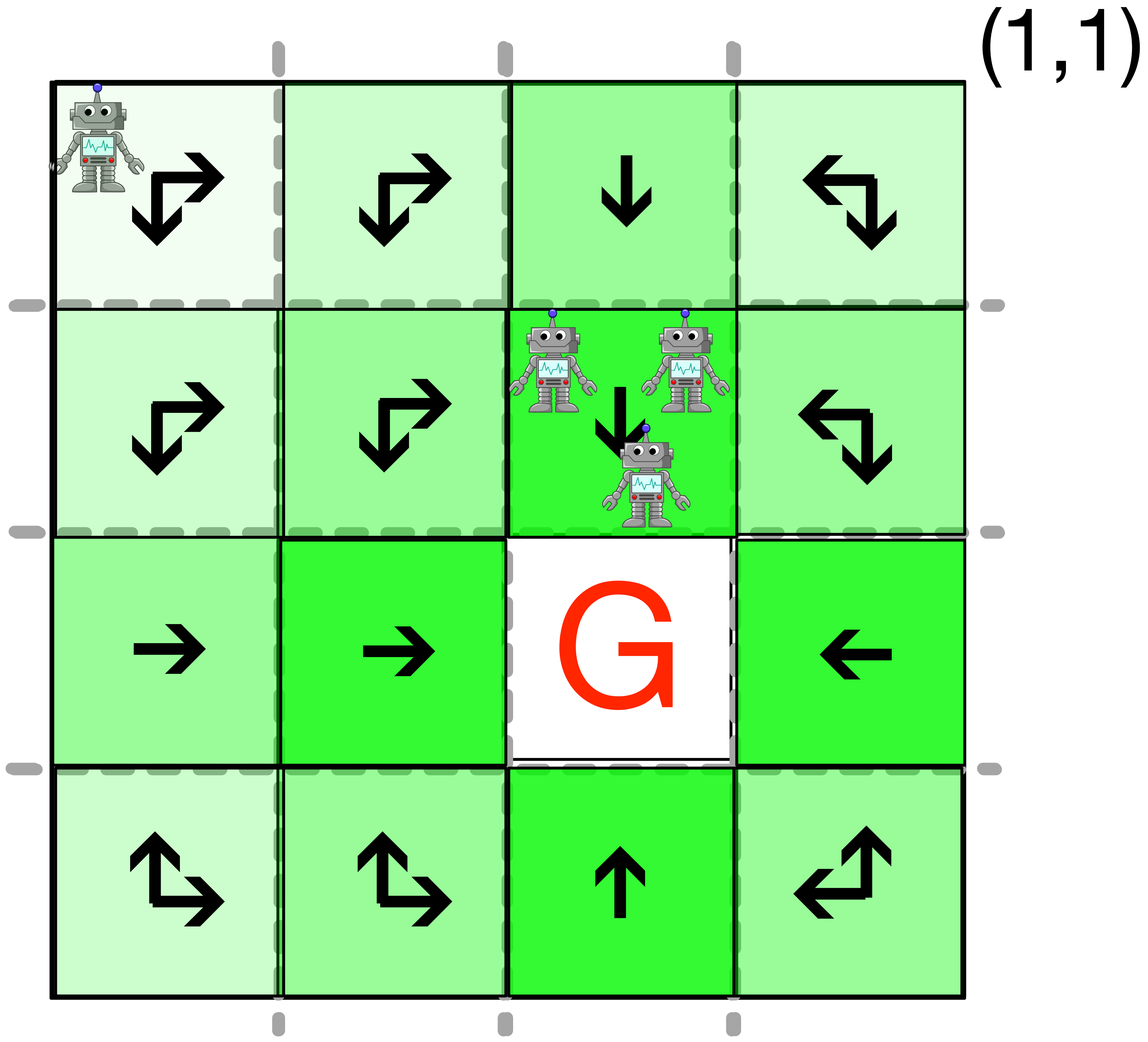
Or here?

 = (.1, .45)



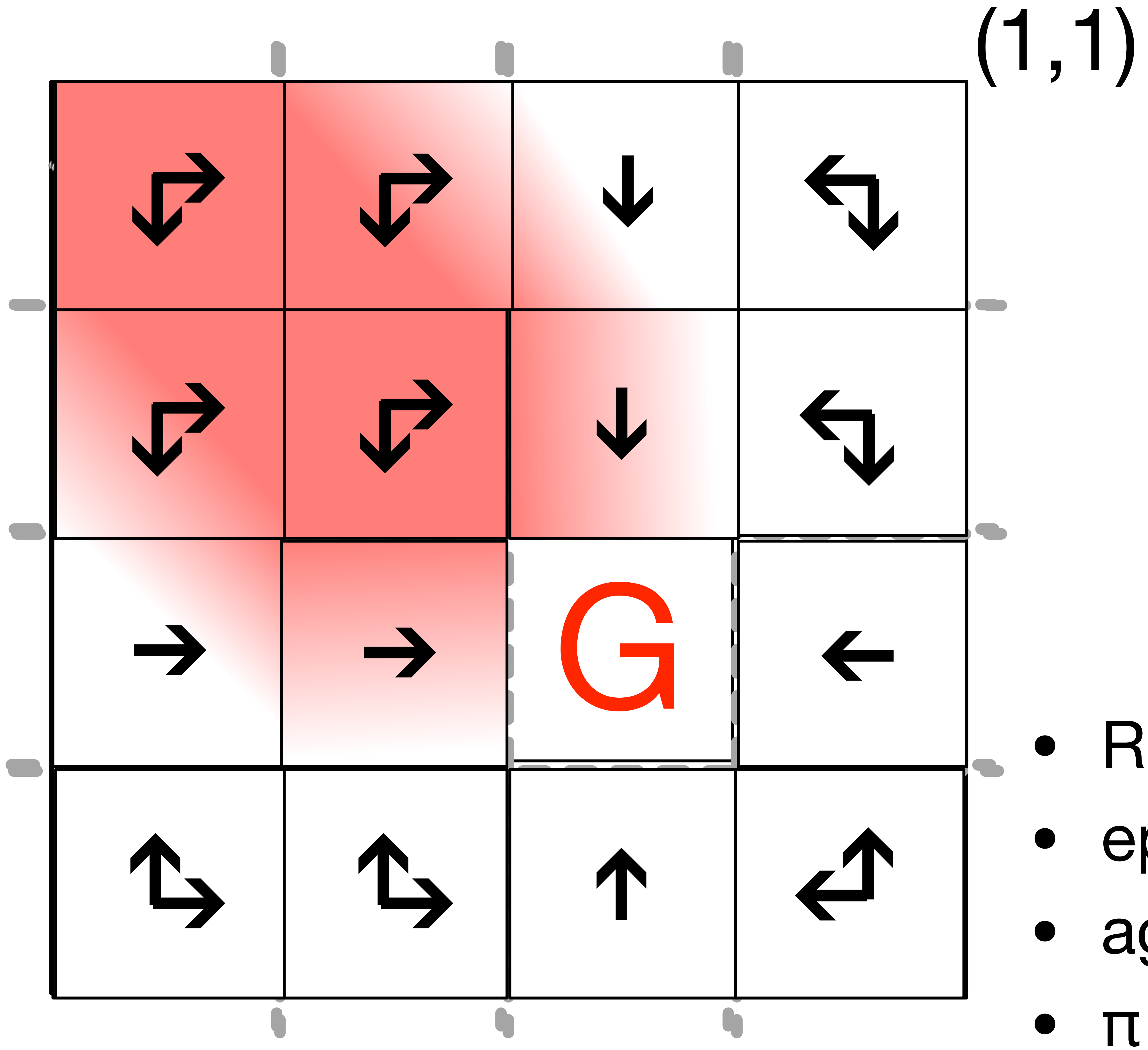
$\boldsymbol{x}(s) =$ $\boldsymbol{x}($$) = \boldsymbol{x}(0.1, 0.45) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

What might the final estimate of the value function look like with this state aggregation?



- R = +1 per step
- episodic, gamma = 1
- agent starts in the top left corner
- π = shortest path policy
- what should $\hat{v}(s, \mathbf{w})$ look like?

What might the μ (proportion of time the agent spends in each state) look like with this state aggregation?



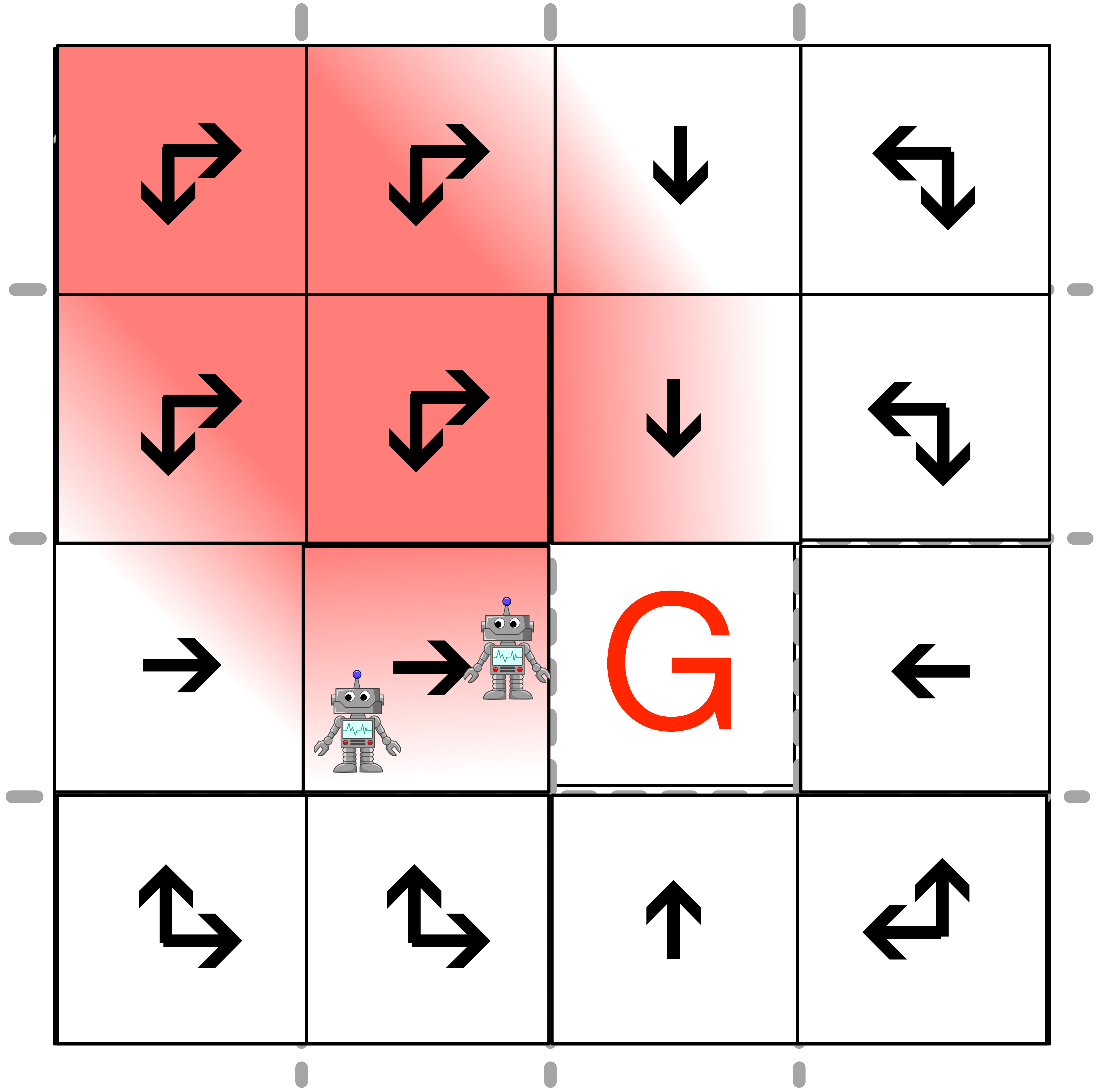
- R = +1 per step
- episodic, gamma = 1
- agent starts in the top left corner
- π = shortest path policy

Mean Squared Value Error

$$\sum_s \mu(s)[v_\pi(s) - \hat{v}(s, \mathbf{w})]^2$$

The fraction of time we spend in S when following policy π

μ(s) Impacts how we update $\hat{v}(s, \mathbf{w})$



Mean Squared Value Error

$$\sum_s \mu(s)[v_\pi(s) - \hat{v}(s, \mathbf{w})]^2$$

The fraction of time we spend in S when following policy π

The usual recipe for gradient descent

1. Specify a function approximation architecture (parametric form of value function)

2. Write down your objective function

3. Take the derivative of objective function with respect to the weights

4. Simplify general gradient expression for your parametric form

5. Make a weight update rule:

   - $\mathbf{w} = \mathbf{w} - \alpha$ gradient

gradient descent

lets try out the recipe

1. Specify a function approximation architecture (parametric form of value function)

- We will use **State Aggregation**
  - so the **features** are always **binary** with only a single active feature that is not zero
  - the value function is a **linear function**
    - that is, we query the value function by a simple procedure:
      1. **query the features** for the current state
      2. take the inner product between the features and the weights

$$v_\pi(s) \approx \hat{v}(s, \mathbf{w}) \doteq \mathbf{w}^\top \mathbf{x}(s) \doteq \sum_{i=1}^{n} w_i \cdot x_i(s)$$

2. Write down your objective function

- We will use the value error

$$\overline{VE}(\mathbf{w}) \doteq \sum_{s \in \mathcal{S}} \mu(s)\left[v_\pi(s) - \hat{v}(s, \mathbf{w})\right]^2$$

$$= \sum_{s \in \mathcal{S}} \mu(s)\left[v_\pi(s) - \boxed{\mathbf{w}^T \mathbf{x}(s)}\right]^2$$

state aggregation

3. Take the gradient of objective function with respect to the weights

$$\nabla \overline{VE}(\mathbf{w}) = \nabla \sum_{s \in \mathcal{S}} \mu(s)[v_\pi(s) - \mathbf{w}^T \mathbf{x}(s)]^2$$

3. Take the gradient of objective function with respect to the weights

$$\nabla \overline{VE}(\mathbf{w}) = \nabla \sum_{s \in \mathcal{S}} \mu(s)[v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)]^2$$

$$= \sum_{s \in \mathcal{S}} \mu(s) \nabla [v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)]^2$$

$$= - \sum_{s \in \mathcal{S}} \mu(s) 2[v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)] \nabla \mathbf{w}^T\mathbf{x}(s)$$

4. Simplify the general gradient expression to be specific for your parametric form

$$\nabla \mathbf{w}^T \mathbf{x}(s) = \mathbf{x}(s)$$

The gradient of the inner product is just **x**

4. Simplify general gradient ...

gradient of
linear value function
approximation (state agg.)

$$\nabla \overline{VE}(\mathbf{w}) = \nabla \sum_{s \in \mathcal{S}} \mu(s)[v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)]^2$$

$$= \sum_{s \in \mathcal{S}} \mu(s) \nabla [v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)]^2$$

$$= -\sum_{s \in \mathcal{S}} \mu(s) 2[v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)] \nabla \mathbf{w}^T\mathbf{x}(s)$$

$$= -\sum_{s \in \mathcal{S}} \mu(s) 2[v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)] \boxed{\mathbf{x}(s)}$$

5. Make weight update rule: **w** = **w** - α gradient

$$\nabla \overline{VE}(\mathbf{w}) = -\sum_{s \in \mathcal{S}} \mu(s) 2[v_\pi(s) - \mathbf{w}^T \mathbf{x}(s)]\mathbf{x}(s)$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha 2[v_\pi(s) - \mathbf{w}^T \mathbf{x}(s)]\mathbf{x}(s)$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha[v_\pi(s) - \mathbf{w}^T \mathbf{x}(s)]\mathbf{x}(s)$$

Wait, Wait, Wait!! We don't have v_π

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha[v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)]\mathbf{x}(s)$$

Let's replace it with something we do have!

Let's replace v_π with something we do have!

Let's call it's replacement Ut

Whatever we use in place of v_π, it should satisfy one criteria!

$$v_\pi(s) = \mathbb{E}[U_t]$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha[v_\pi(s) - \mathbf{w}^T\mathbf{x}(s)]\mathbf{x}(s)$$

Whatever we use in place of v_π, it should satisfy one criteria!

$$v_\pi(s) = \mathbb{E}[U_t]$$

We know one such replacement, that meets this criteria!

$$U_t \doteq G_t$$

A sample of the return!!

Since we are using sample returns we have a Monte Carlo algorithm!

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha[G_t - \mathbf{w}^T\mathbf{x}(s)]\mathbf{x}(s)$$

Monte Carlo Policy Evaluation for finding v_π