Partial Observations

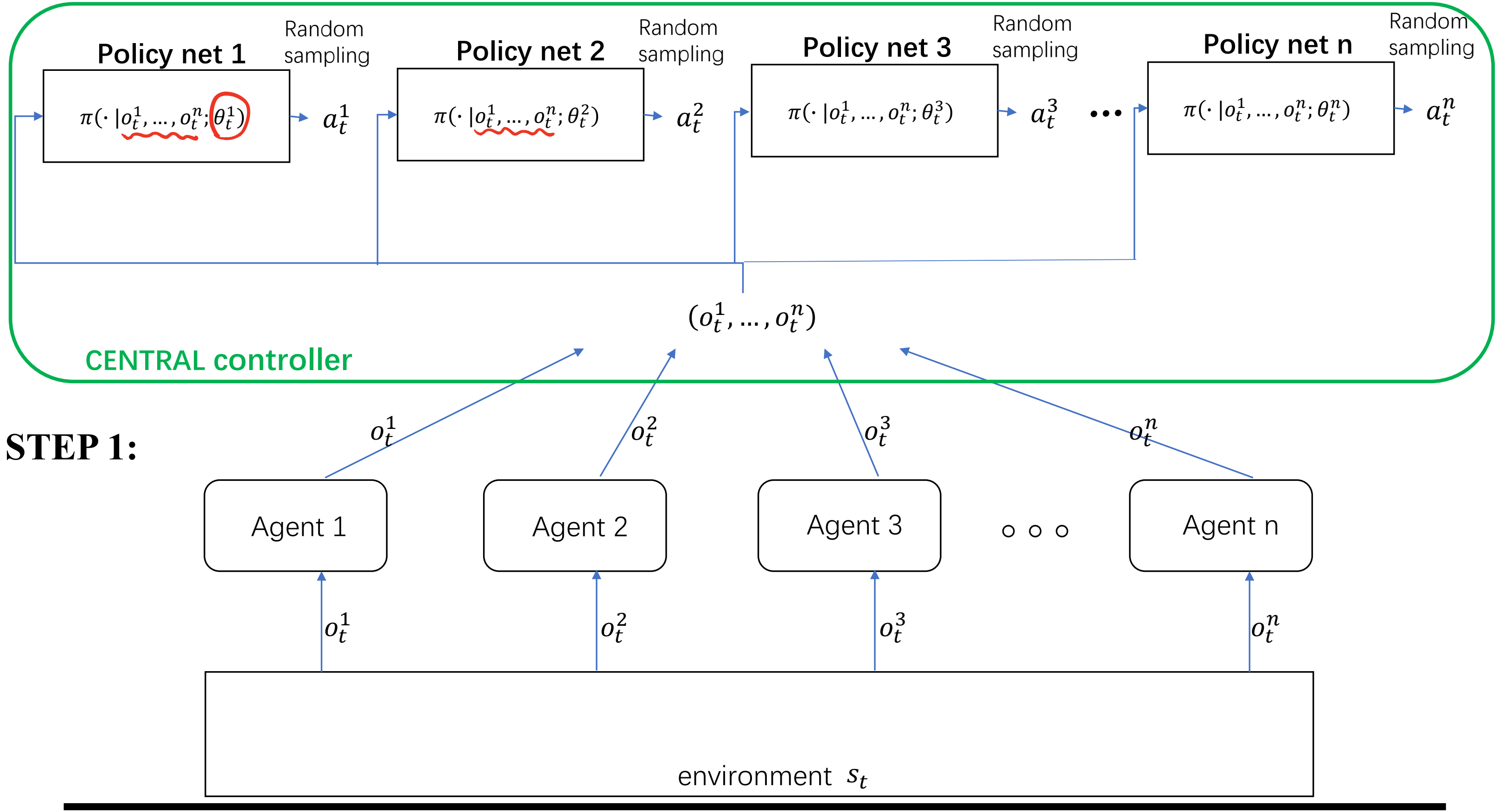
不同agents在一个时刻具有相同的状态，但agent不一定能完整的观测到。

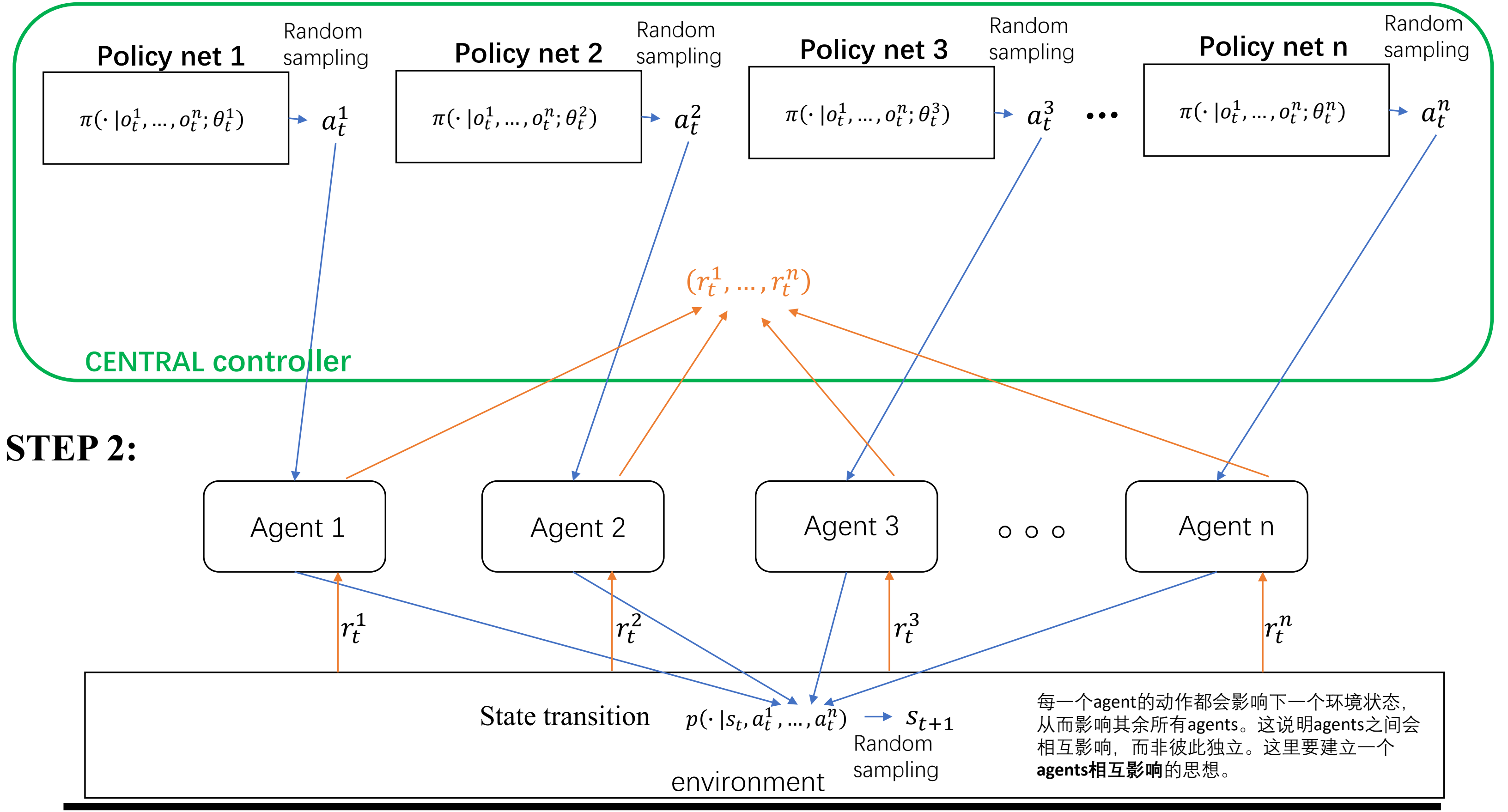
- An agent may or may not have full knowledge of the state, s.
- Let o^i be the i-th agent's observation.
- Partial observation: $o^i \neq s$.
- Full observation: $o^1 = \cdots = o^n = s$.

Architecture 1:

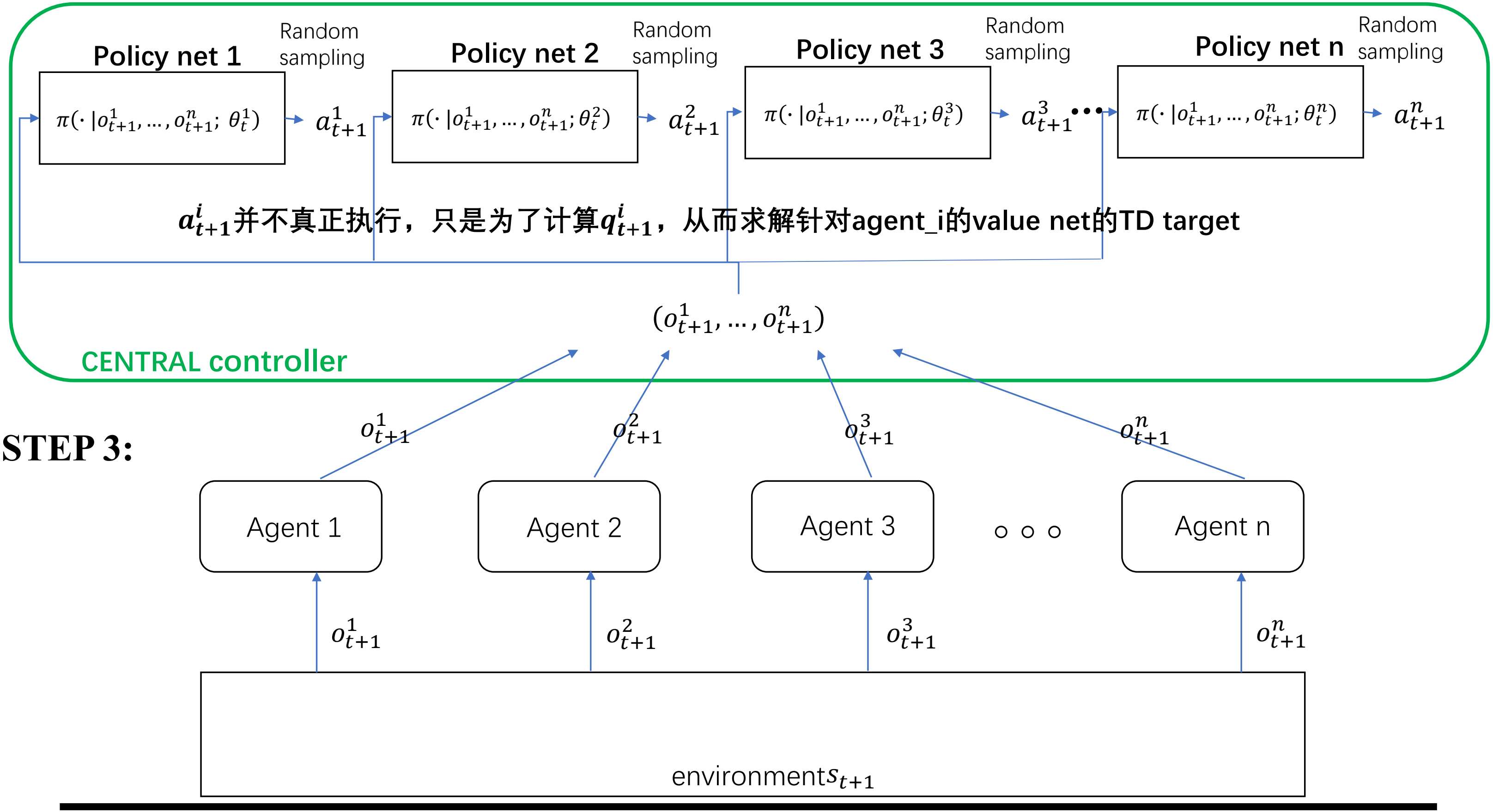
Centralized Actor-Critic Method

策略网络全在中央，训练在中央做，即，控制器用所有的观测$(o_t^1,\ldots,o_t^n)$来训练各个策略网络，这种训练方式叫centralized training。

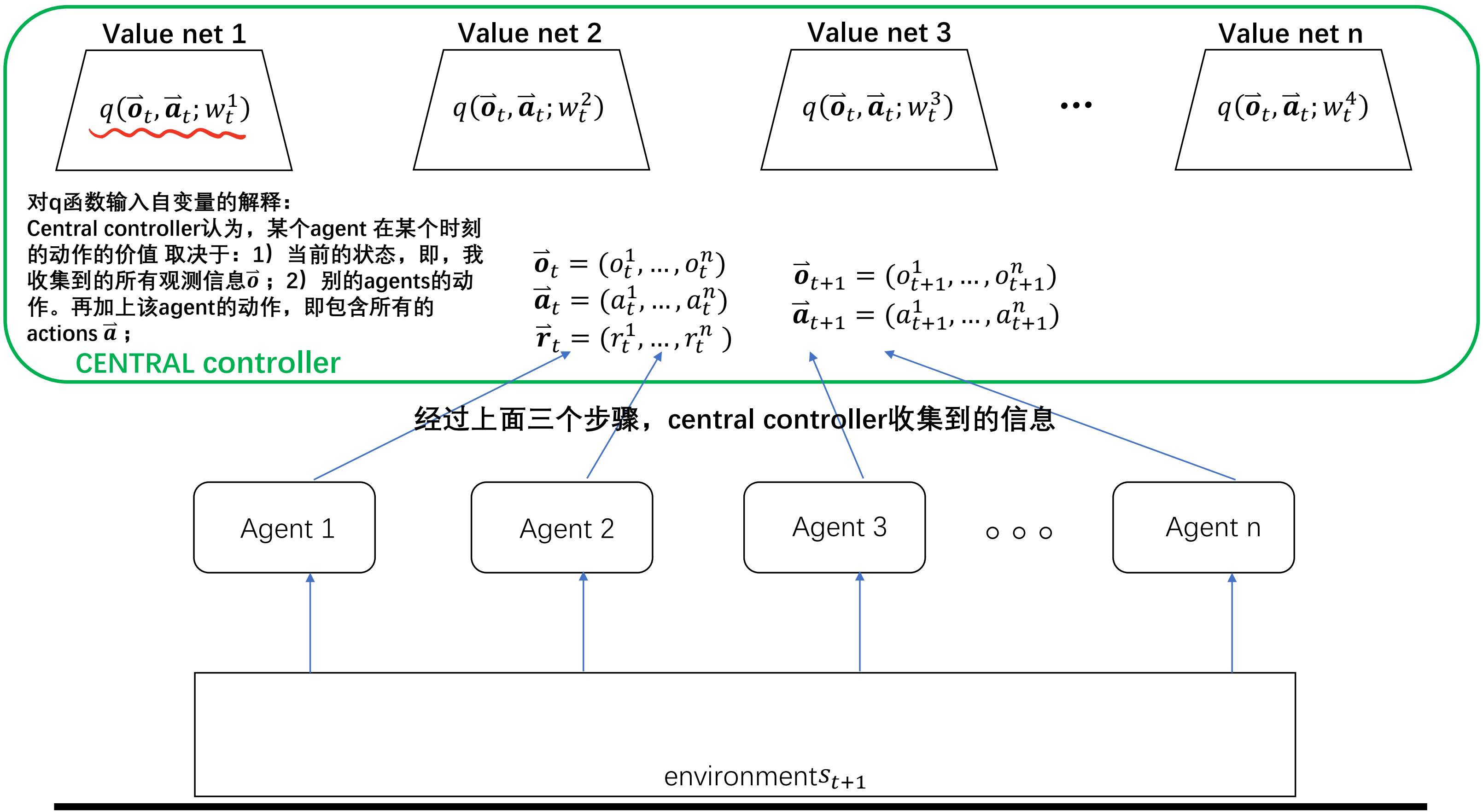



Policy net 1
Random sampling
π(· |o_t^1, …, o_t^n; θ_t^1)
a_t^1
Policy net 2
Random sampling
π(· |o_t^1, …, o_t^n; θ_t^2)
a_t^2
Policy net 3
Random sampling
π(· |o_t^1, …, o_t^n; θ_t^3)
a_t^3
Policy net n
Random sampling
π(· |o_t^1, …, o_t^n; θ_t^n)
a_t^n
(r_t^1, …, r_t^n)
CENTRAL controller
STEP 2:
Agent 1
Agent 2
Agent 3
Agent n
r_t^1
r_t^2
r_t^3
r_t^n
State transition
p(· |s_t, a_t^1, …, a_t^n)
s_{t+1}
Random sampling
environment
每一个agent的动作都会影响下一个环境状态，从而影响其余所有agents。这说明agents之间会相互影响，而非彼此独立。这里要建立一个agents相互影响的思想。

STEP 3:



Policy net 1
Random sampling
$\pi(\cdot\,|o_{t+1}^1, \dots, o_{t+1}^n;\ \theta_t^1)$
a_{t+1}^1
Policy net 2
Random sampling
$\pi(\cdot\,|o_{t+1}^1, \dots, o_{t+1}^n;\theta_t^2)$
a_{t+1}^2
Policy net 3
Random sampling
$\pi(\cdot\,|o_{t+1}^1, \dots, o_{t+1}^n;\theta_t^3)$
a_{t+1}^3
Policy net n
Random sampling
$\pi(\cdot\,|o_{t+1}^1, \dots, o_{t+1}^n;\theta_t^n)$
a_{t+1}^n
a_{t+1}^i并不真正执行，只是为了计算q_{t+1}^i，从而求解针对agent_i的value net的TD target
$(o_{t+1}^1, \dots, o_{t+1}^n)$
CENTRAL controller
o_{t+1}^1
o_{t+1}^2
o_{t+1}^3
o_{t+1}^n
Agent 1
Agent 2
Agent 3
Agent n
o_{t+1}^1
o_{t+1}^2
o_{t+1}^3
o_{t+1}^n
environments_{t+1}


Value net 1
$q(\vec{o}_t, \vec{a}_t; w_t^1)$
Value net 2
$q(\vec{o}_t, \vec{a}_t; w_t^2)$
Value net 3
$q(\vec{o}_t, \vec{a}_t; w_t^3)$
Value net n
$q(\vec{o}_t, \vec{a}_t; w_t^4)$
对q函数输入自变量的解释：
Central controller认为，某个agent 在某个时刻的动作的价值 取决于：1）当前的状态，即，我收集到的所有观测信息$\vec{o}$；2）别的agents的动作。再加上该agent的动作，即包含所有的actions $\vec{a}$；
$\vec{o}_t = (o_t^1, \dots, o_t^n)$
$\vec{a}_t = (a_t^1, \dots, a_t^n)$
$\vec{r}_t = (r_t^1, \dots, r_t^n)$
$\vec{o}_{t+1} = (o_{t+1}^1, \dots, o_{t+1}^n)$
$\vec{a}_{t+1} = (a_{t+1}^1, \dots, a_{t+1}^n)$
CENTRAL controller
经过上面三个步骤，central controller收集到的信息
Agent 1
Agent 2
Agent 3
Agent n
environments_{t+1}

- Centralized Training: Training is performed by the controller.
  - The controller knows all the observations, actions, and rewards.
  - Train $\pi\left(a_t^i \middle| \mathbf{o}_t; \boldsymbol{\theta}_t^i\right)$ using policy gradient.
  - Train $q\left(\mathbf{o}_t, \mathbf{a}_t; \mathbf{w}_t^i\right)$ using TD algorithm.

都要用到 $\begin{cases} \vec{\boldsymbol{o}}_t = (o_t^1, \dots, o_t^n) \\ \vec{\boldsymbol{a}}_t = (a_t^1, \dots, a_t^n) \\ \vec{\boldsymbol{r}}_t = (r_t^1, \dots, r_t^n) \\ \vec{\boldsymbol{o}}_{t+1} = (o_{t+1}^1, \dots, o_{t+1}^n) \\ \vec{\boldsymbol{a}}_{t+1} = (a_{t+1}^1, \dots, a_{t+1}^n) \end{cases}$

STEP 4:

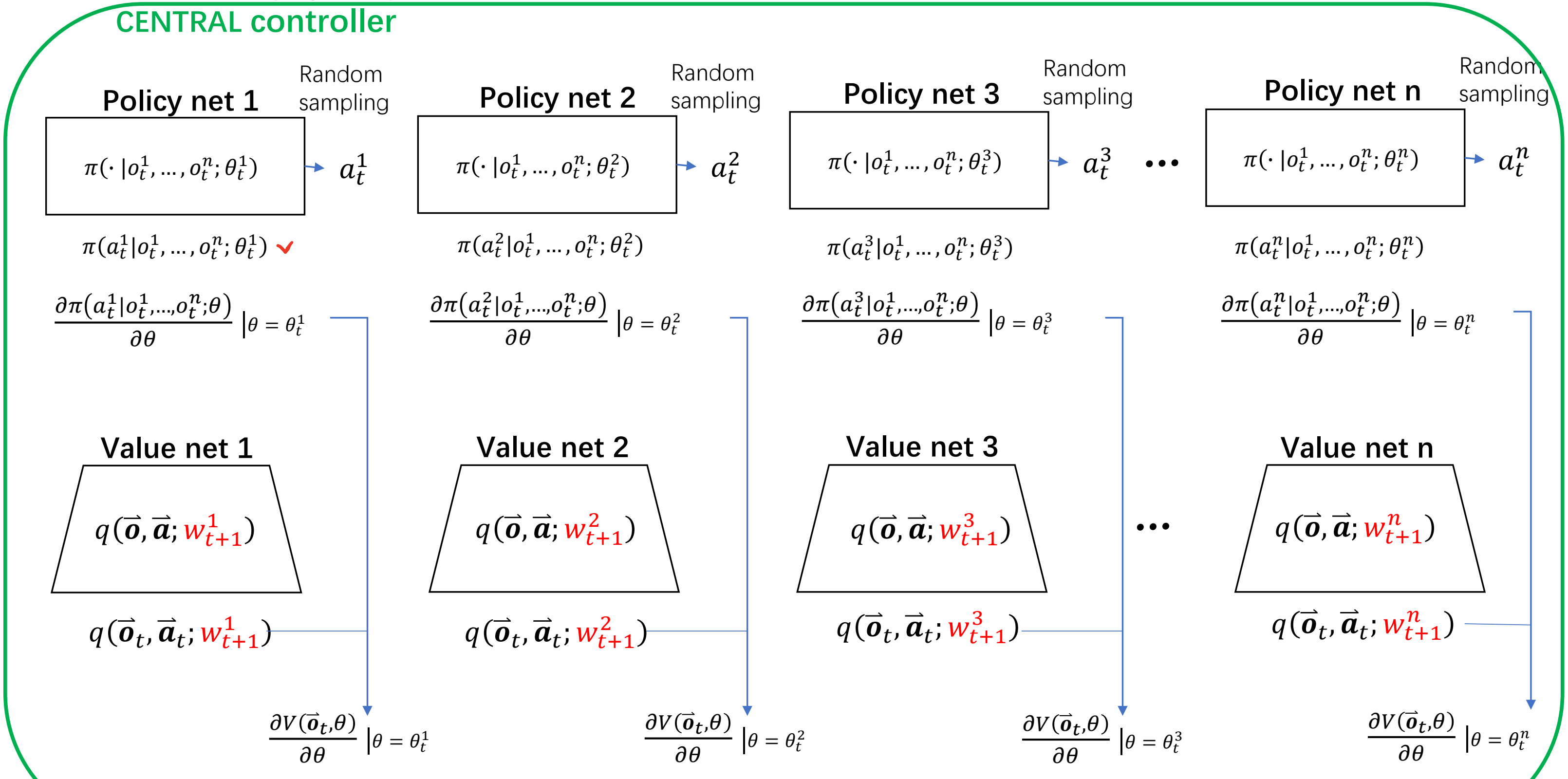
Train $q\left(\mathbf{o}_t, \mathbf{a}_t; \mathbf{w}_t^i\right)$ using TD algorithm.

	Value net 1	Value net 2	Value net 3	…	Value net n
	$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^1)$	$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^2)$	$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^3)$		$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^4)$
Pred:	$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^1)$ ✓	$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^2)$	$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^3)$		$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^4)$
TD target:	$r_t^1 + q(\vec{\boldsymbol{o}}_{t+1}, \vec{\boldsymbol{a}}_{t+1}; w_t^1)$ ✓	$r_t^2 + q(\vec{\boldsymbol{o}}_{t+1}, \vec{\boldsymbol{a}}_{t+1}; w_t^2)$	$r_t^3 + q(\vec{\boldsymbol{o}}_{t+1}, \vec{\boldsymbol{a}}_{t+1}; w_t^3)$		$r_t^4 + q(\vec{\boldsymbol{o}}_{t+1}, \vec{\boldsymbol{a}}_{t+1}; w_t^4)$

由此计算各个value net的TD error，利用梯度下降更新各自的权重，例如$w_t^1 \rightarrow w_{t+1}^1$

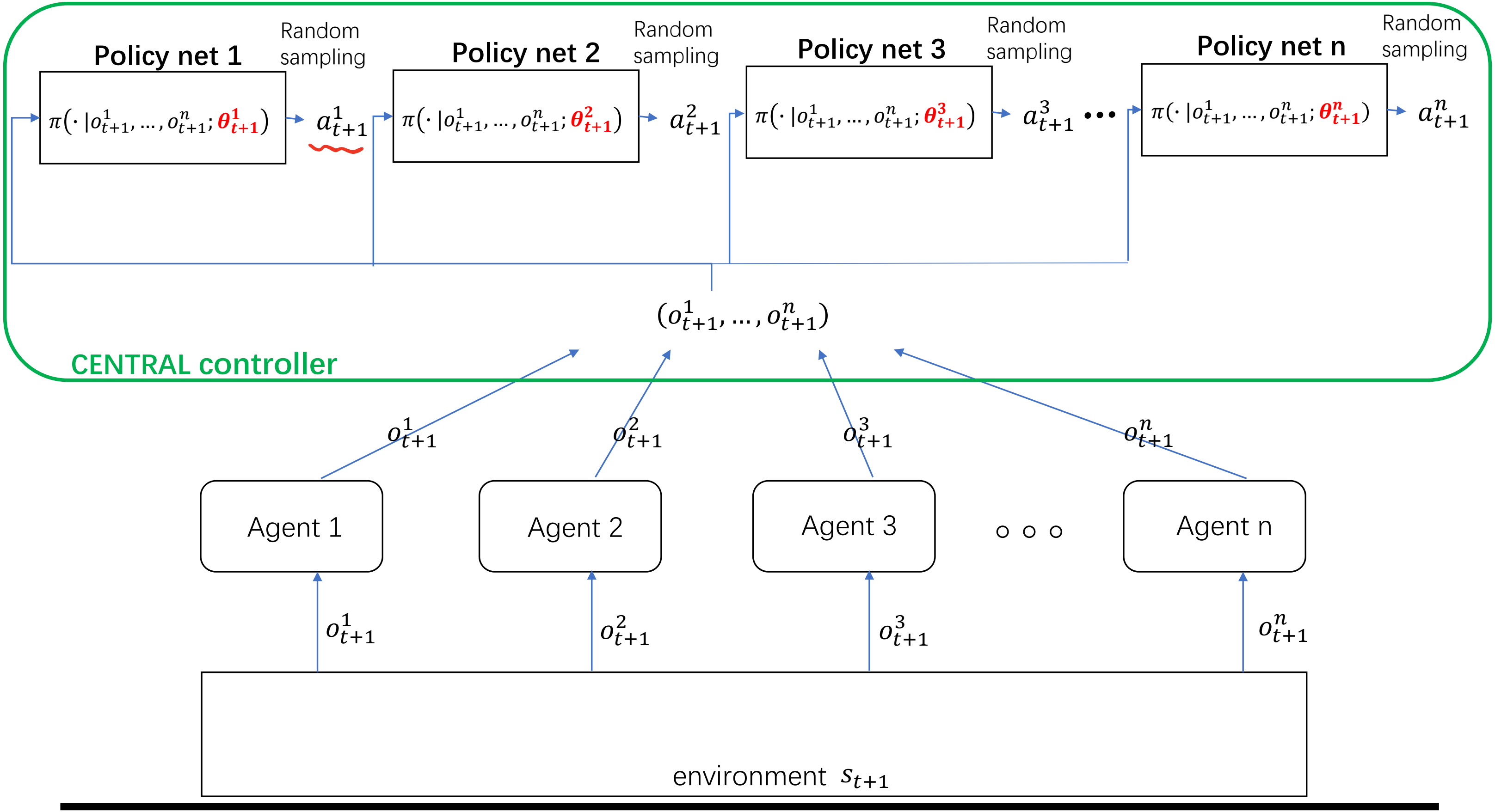
CENTRAL controller

STEP 5: Train $\pi\left(a_t^i \mid \mathbf{o}_t; \boldsymbol{\theta}_t^i\right)$ using policy gradient.



利用各个policy gradient来更新对应的policy net，例如$\theta_t^1 \rightarrow \theta_{t+1}^1$

t+1时刻的训练开始，循环往复…



- Centralized Execution: Decisions are made by the controller.
  - For all i, the i-th agent sends its observation, o^i, to the controller.
  - The controller knows $\mathbf{o} = [o^1, o^2, \cdots, o^n]$.
  - For all i, the controller samples action by $a^i \sim \pi(\cdot \mid \mathbf{o}\,; \boldsymbol{\theta}^i)$ and sends a^i to the i-th agent.

中央控制器上训练出n个策略网络，结构可以相同，参数可能不同。

一个agent只能知道自己的观测，没有足够的信息做决策。所以策略网络不能部署在agent上。

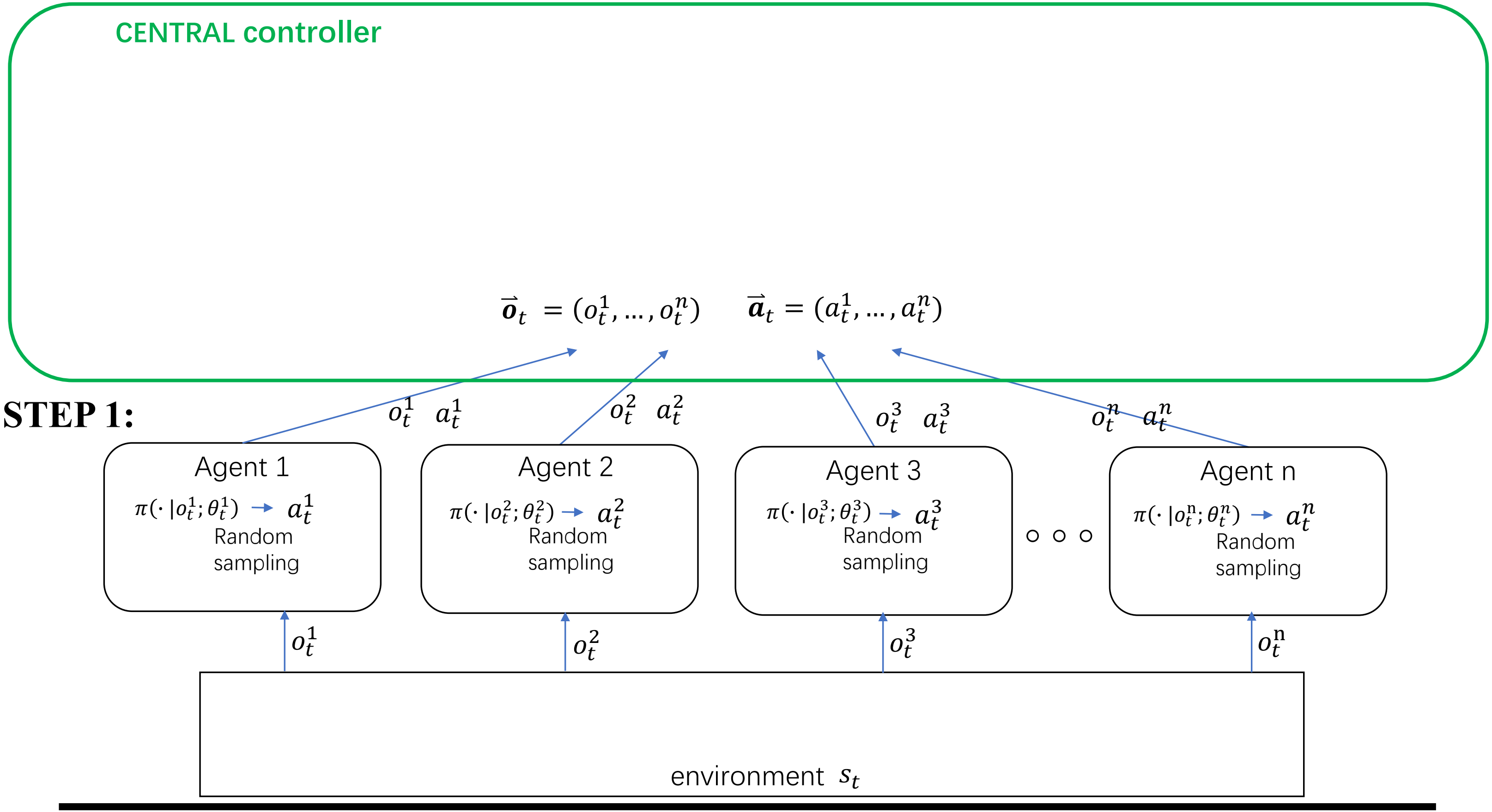
实际执行时，汇报所有观测到中央，中央根据全局信息做出决策，告诉每个agent应该做什么。

Shortcoming: Slow during Execution

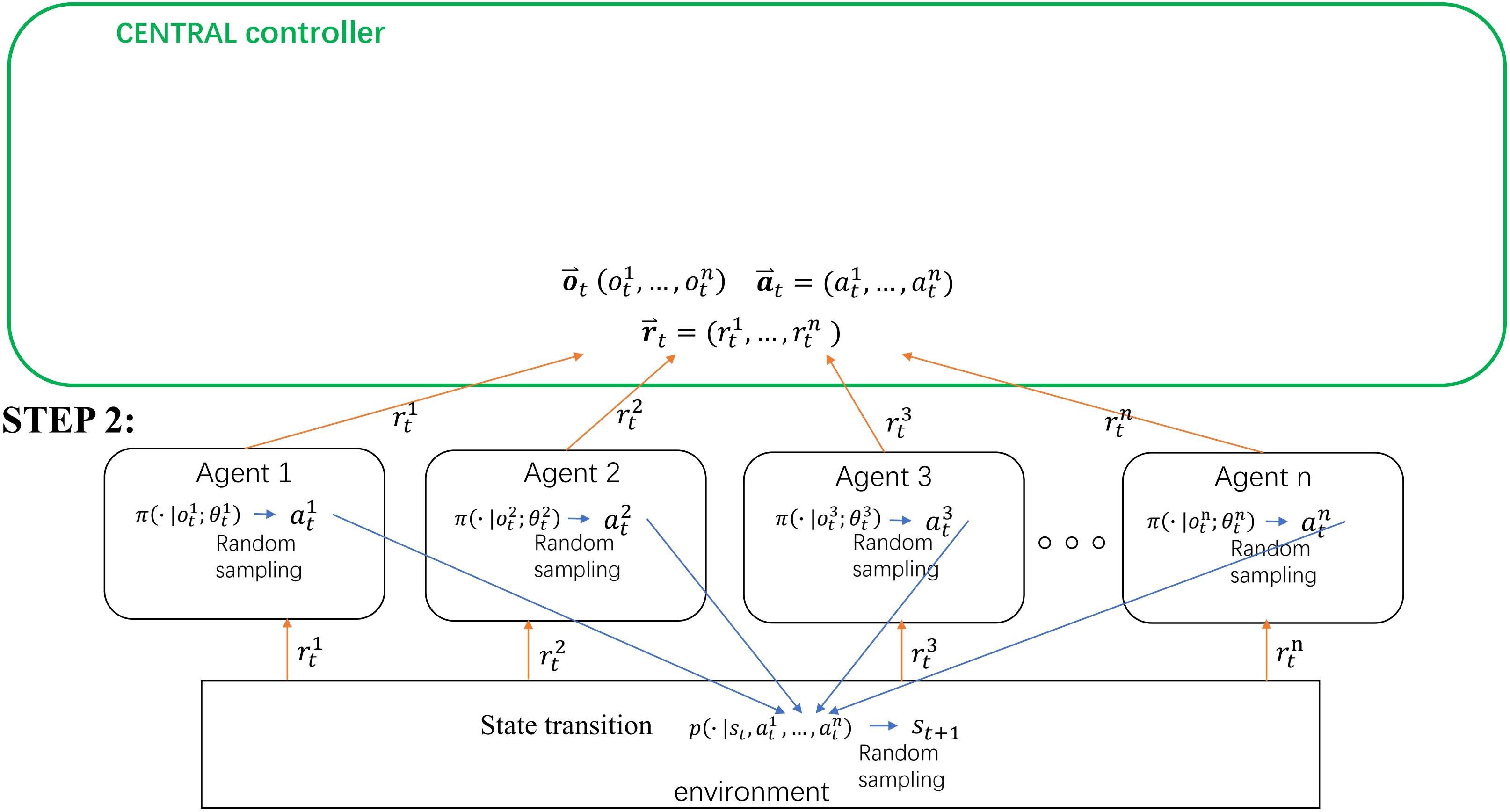
- All the agents send their observations to the central controller.
- The central controller makes decisions, $\mathbf{a} = [a^1, a^2, \cdots, a^n]$, and sends a^i to the i-th agent.
- Communication and synchronization cost time.
- Real-time decision is impossible.

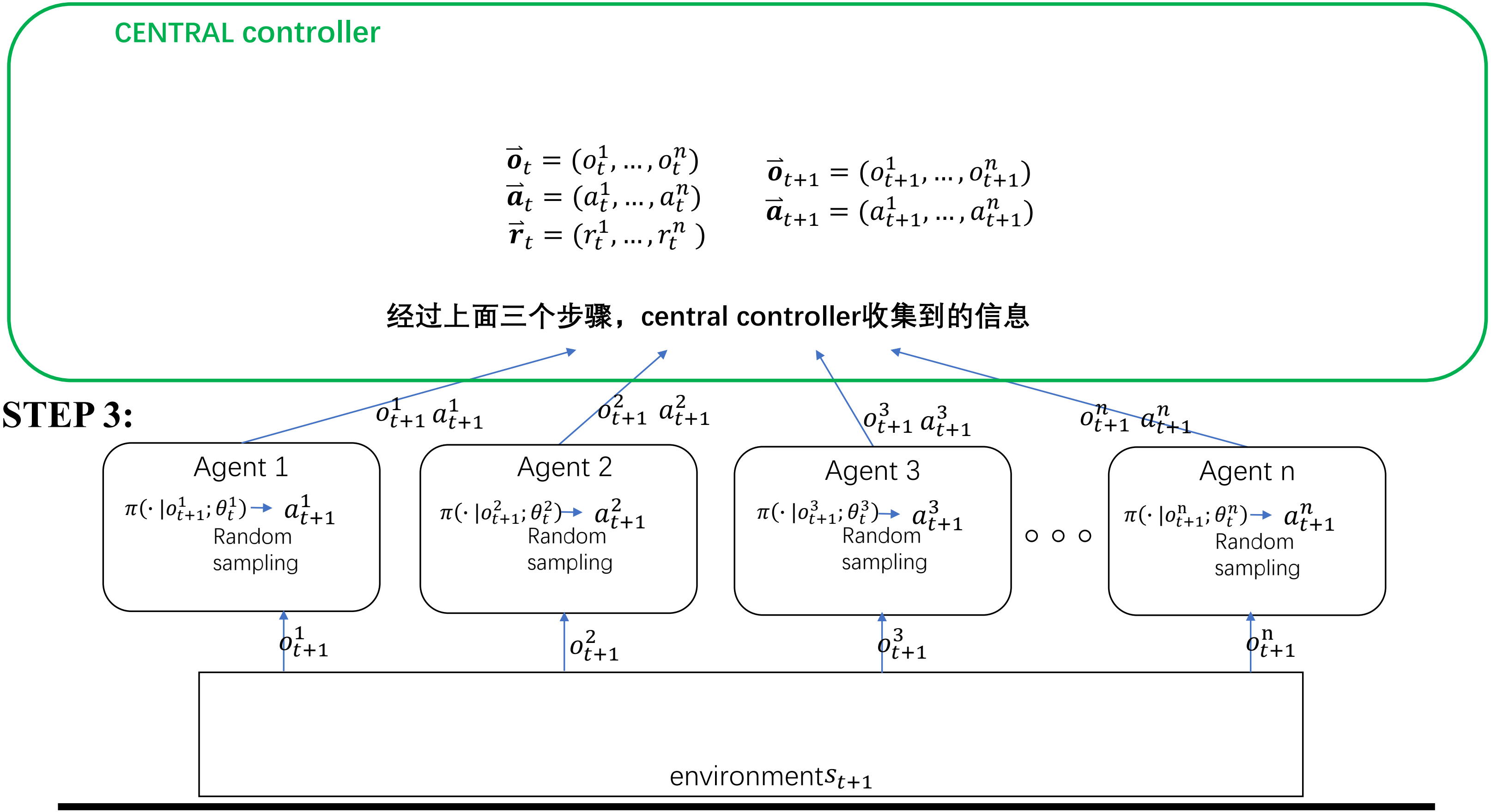
Architecture 2:

Centralized Training with Decentralized Execution


CENTRAL controller
$\vec{\boldsymbol{o}}_t = (o_t^1, \dots, o_t^n)$
$\vec{\boldsymbol{a}}_t = (a_t^1, \dots, a_t^n)$
STEP 1:
o_t^1 a_t^1
o_t^2 a_t^2
o_t^3 a_t^3
o_t^n a_t^n
Agent 1
$\pi(\cdot\,|o_t^1;\theta_t^1) \rightarrow a_t^1$
Random sampling
Agent 2
$\pi(\cdot\,|o_t^2;\theta_t^2) \rightarrow a_t^2$
Random sampling
Agent 3
$\pi(\cdot\,|o_t^3;\theta_t^3) \rightarrow a_t^3$
Random sampling
Agent n
$\pi(\cdot\,|o_t^n;\theta_t^n) \rightarrow a_t^n$
Random sampling
o_t^1
o_t^2
o_t^3
o_t^n
environment s_t

STEP 2:


CENTRAL controller
$\vec{\boldsymbol{o}}_t\ (o_t^1, \ldots, o_t^n)$ $\vec{\boldsymbol{a}}_t = (a_t^1, \ldots, a_t^n)$
$\vec{\boldsymbol{r}}_t = (r_t^1, \ldots, r_t^n)$
r_t^1
r_t^2
r_t^3
r_t^n
Agent 1
$\pi(\cdot\,|o_t^1;\theta_t^1) \rightarrow a_t^1$
Random sampling
Agent 2
$\pi(\cdot\,|o_t^2;\theta_t^2) \rightarrow a_t^2$
Random sampling
Agent 3
$\pi(\cdot\,|o_t^3;\theta_t^3) \rightarrow a_t^3$
Random sampling
Agent n
$\pi(\cdot\,|o_t^n;\theta_t^n) \rightarrow a_t^n$
Random sampling
r_t^1
r_t^2
r_t^3
r_t^n
State transition
$p(\cdot\,|s_t, a_t^1, \ldots, a_t^n) \rightarrow s_{t+1}$
Random sampling
environment


CENTRAL controller
$\vec{\boldsymbol{o}}_t = (o_t^1, \dots, o_t^n)$
$\vec{\boldsymbol{a}}_t = (a_t^1, \dots, a_t^n)$
$\vec{\boldsymbol{r}}_t = (r_t^1, \dots, r_t^n)$
$\vec{\boldsymbol{o}}_{t+1} = (o_{t+1}^1, \dots, o_{t+1}^n)$
$\vec{\boldsymbol{a}}_{t+1} = (a_{t+1}^1, \dots, a_{t+1}^n)$
经过上面三个步骤，central controller收集到的信息
STEP 3:
$o_{t+1}^1\ a_{t+1}^1$
$o_{t+1}^2\ a_{t+1}^2$
$o_{t+1}^3\ a_{t+1}^3$
$o_{t+1}^n\ a_{t+1}^n$
Agent 1
$\pi(\cdot\,|o_{t+1}^1;\theta_t^1) \rightarrow a_{t+1}^1$
Random sampling
Agent 2
$\pi(\cdot\,|o_{t+1}^2;\theta_t^2) \rightarrow a_{t+1}^2$
Random sampling
Agent 3
$\pi(\cdot\,|o_{t+1}^3;\theta_t^3) \rightarrow a_{t+1}^3$
Random sampling
Agent n
$\pi(\cdot\,|o_{t+1}^n;\theta_t^n) \rightarrow a_{t+1}^n$
Random sampling
o_{t+1}^1
o_{t+1}^2
o_{t+1}^3
o_{t+1}^n
environments_{t+1}

STEP 4:

Train $q(\mathbf{o}_t, \mathbf{a}_t; \mathbf{w}_t^{i})$ using TD algorithm. 与fully centralized的value net training是一样的

CENTRAL controller

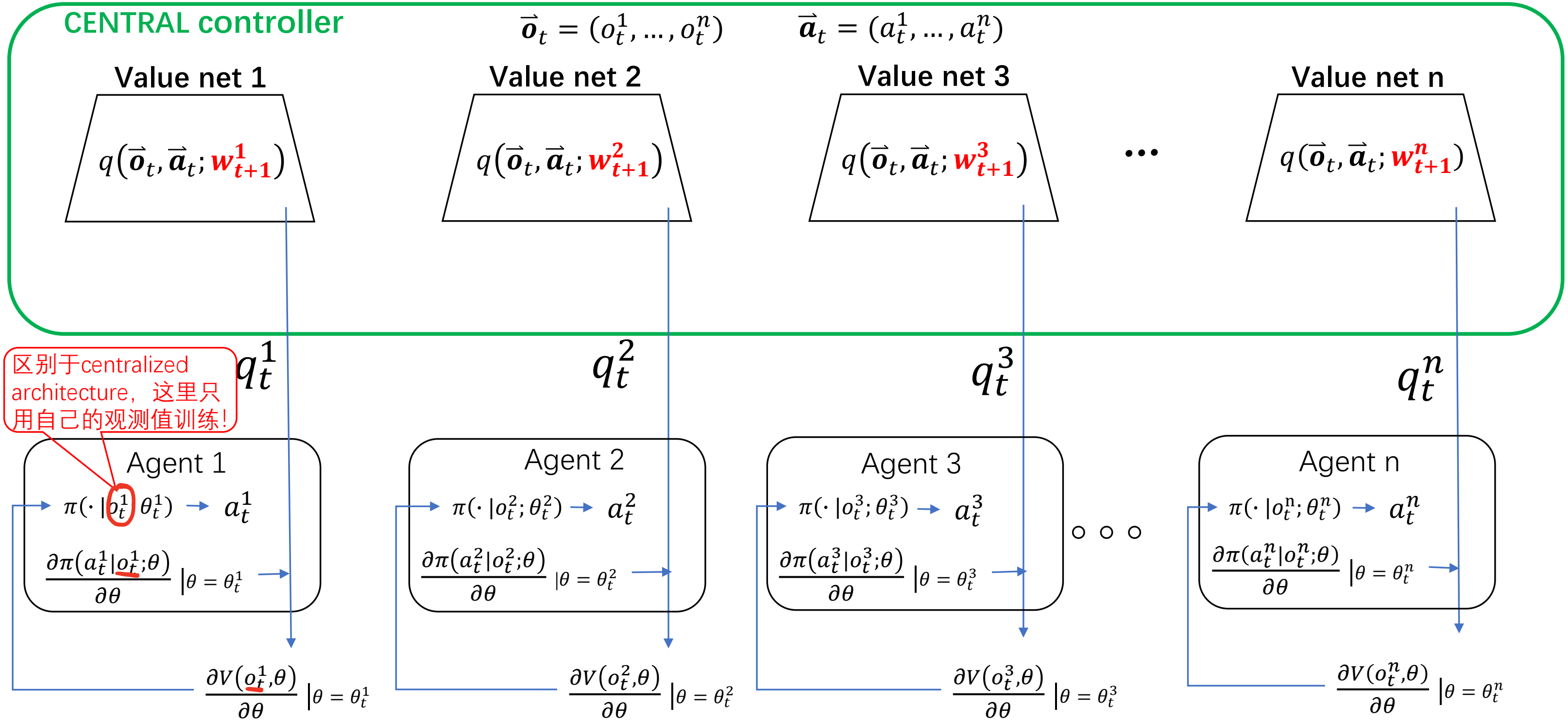
Value net 1	Value net 2	Value net 3	…	Value net n
$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^1)$	$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^2)$	$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^3)$		$q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^4)$

Pred: $q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^1)$ $\quad q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^2)$ $\quad q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^3)$ $\quad q(\vec{\boldsymbol{o}}_t, \vec{\boldsymbol{a}}_t; w_t^4)$

TD target: $r_t^1 + q(\vec{\boldsymbol{o}}_{t+1}, \vec{\boldsymbol{a}}_{t+1}; w_t^1)$ $\quad r_t^2 + q(\vec{\boldsymbol{o}}_{t+1}, \vec{\boldsymbol{a}}_{t+1}; w_t^2)$ $\quad r_t^3 + q(\vec{\boldsymbol{o}}_{t+1}, \vec{\boldsymbol{a}}_{t+1}; w_t^3)$ $\quad r_t^4 + q(\vec{\boldsymbol{o}}_{t+1}, \vec{\boldsymbol{a}}_{t+1}; w_t^4)$

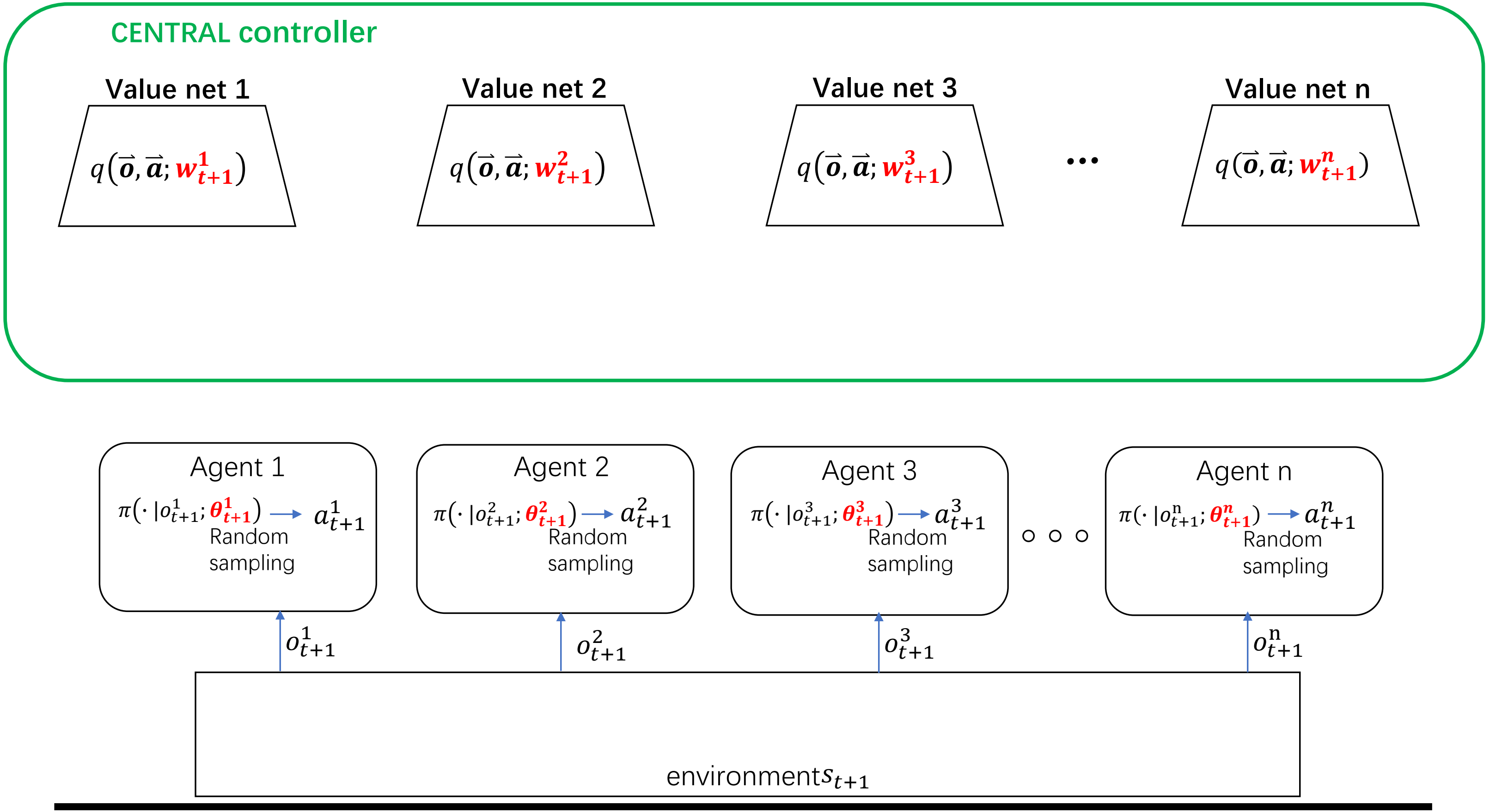
由此计算各个value net的TD error，并更新各自的权重，例如$w_t^1 \rightarrow w_{t+1}^1$

STEP 5: Each agent locally trains the actor, $\pi\left(a_t^i \middle| o_t^i\,;\boldsymbol{\theta}_t^i\right)$, using policy gradient.



利用各个policy gradient来更新对应的policy net，例如$\theta_t^1 \rightarrow \theta_{t+1}^1$

t+1时刻的训练开始，循环往复…



Decentralized Execution

中心化训练，去中心化执行：每个agent都有自己的策略网络。训练的时候需要一个中央控制器它帮助agents训练value nets，从而帮助训练policy net。

结束训练之后，就不需要中央控制器了。每个agent独立跟环境交互，用自己的策略网络、基于自己局部的观测来做决策（不依赖别人的观测）。

Architecture 3:

Fully Decentralized

Fully Decentralized Actor-Critic Method

- The i-th agent has a policy network (actor): $\pi\left(a^i \middle| o^i; \boldsymbol{\theta}^i\right)$.
- The i-th agent has a value network (critic): $q\left(\boxed{o^i}, \boxed{a^i}; \mathbf{w}^i\right)$.
- Agents do not share observations and actions.
- Train the policy and value networks in the same way as the single-agent setting.
- This does not work well.

该agent的policy确实可以仅采用自身的观测，但是，**在multi-agent交互环境中，agent的动作价值必须要考虑全局观测（所有agents的观测总和）、其余agents的动作。**然而，这种去中心化的方式并没有考虑这一点，而仅仅采用自身的观测来得到action value，这是不合理的。这就好比在一个互联的世界中，仅仅通过自己片面的认识来评价自己的行为，显然得不出好的行为准则。也就是说，训练出来的policy并不能获得最好的效果。这就是fully decentralized的缺点。

Architecture 的比较

	Policy (Actor)	Value (Critic)
Fully Decentralized	$\pi(a^i \mid o^i; \boldsymbol{\theta}^i)$	$q(o^i, a^i; \mathbf{w}^i)$

- The agents are independent.
- One agent is unaware of the other agents' observations and actions.
- Train every agent in the same way as single-agent RL.
- This does not work well.

Agent 自己做决策，只根据自己的不完全观测 o^i 来做出action。**这在现实环境中是合理的。**

Agent 只根据自己的不完全观测 o^i 来对自己的动作进行打分。**在多agents的环境中，显然不合理。**因为其他的agents也会对你的动作价值有影响。

Fully Centralized	$\pi(a^i \mid \mathbf{o}; \boldsymbol{\theta}^i)$	$q(\mathbf{o}, \mathbf{a}; \mathbf{w}^i)$

- All the policy and value networks are in the central controller.
- Agents send everything to the controller.
- The controller makes decisions based on all the agents' observations. Agents do not make decisions.
- The controller tells every agent what to do.

Central controller从全局的角度出发，利用全局的观测 o，基于各个agents的policy nets，**统筹地**给出各个agent的动作，然后发给agent执行。

Central controller认为：agent的action-value要由当前所有的观测 o 以及其他所有agents的actions(再加上这个agent的action) a 来决定。这种思想，就是在**考虑各个agents之间的以环境为媒介的交互。**

Centralized Training, Decentralized Execution	$\pi(a^i \mid o^i; \boldsymbol{\theta}^i)$	$q(\mathbf{o}, \mathbf{a}; \mathbf{w}^i)$

- Each agent has its own policy network.
- The central controller has all the value networks.
- The central controller helps with the training; it is disabled during execution.

Parameter Sharing?

- Do not share parameters if the agents are non-exchangeable.
- Share parameters if the agents are exchangeable.