1. Il Panorama Generale: Come si collegano?

Immagina questo progetto come un corpo umano:

- **Isaac Lab (L'Ambiente):** È la realtà fisica (simulata) in cui vive il robot. Fornisce la gravità, le collisioni e i sensori.
- **Crazyflie (Il Robot):** Il "corpo" specifico, un piccolo drone con 4 motori.
- **skrl (L'Allenatore):** È la libreria che gestisce l'algoritmo di apprendimento (es. PPO). Decide come "punire" o "premiare" il robot.
- **snnTorch (Il Cervello):** È l'architettura della rete neurale. Invece di una rete neurale classica (ANN), userai neuroni che comunicano tramite impulsi elettrici (spike), simulando meglio il cervello biologico.

2. snnTorch: Reti Neurali a Impulsi

A differenza delle reti tradizionali dove i dati scorrono come valori continui, in **snnTorch** i dati sono binari (0 o 1) nel tempo.

- **Concetto chiave:** Il neurone **LIF (Leaky Integrate-and-Fire)**. Accumula potenziale di membrana nel tempo; quando supera una soglia, emette uno "spike" e si resetta.
- **Perché usarlo?** Le SNN sono estremamente efficienti dal punto di vista energetico se eseguite su hardware neuromorfico e sono naturalmente adatte a gestire dati temporali.

Cosa studiare nella doc:

- Come definire un neurone snn.LIF.
- Il concetto di **Backpropagation Through Time (BPTT)** applicato agli spike.

3. Isaac Lab: Simulazione ad Alte Prestazioni

Isaac Lab (basato su NVIDIA Isaac Sim) è fondamentale perché permette di simulare migliaia di droni contemporaneamente sulla GPU utilizzando **PhysX**.

- **Il Task (Hovering):** Il tuo obiettivo è far sì che il Crazyflie mantenga una posizione stabile a mezz'aria.
- **Vettorizzazione:** Non simulerai un solo drone, ma forse 4096 droni in parallelo. Questo è il motivo per cui l'apprendimento è così veloce.

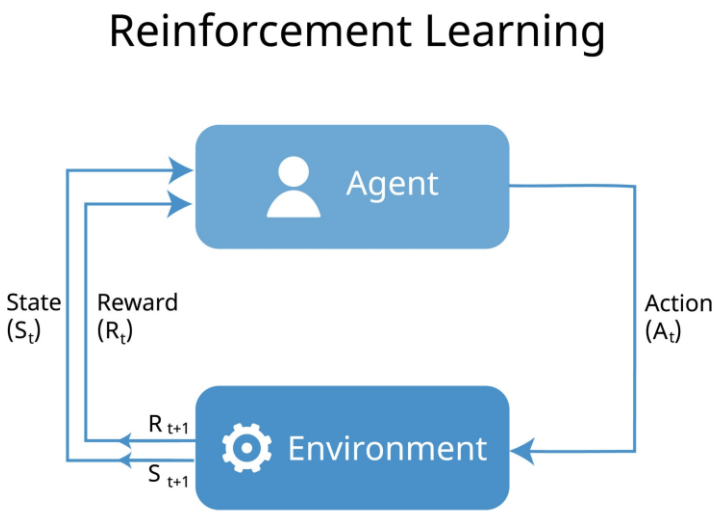
Cosa guardare nel codice del Crazyflie:

- **Observations:** Quali dati riceve il drone? (posizione, orientamento, velocità).

- **Actions:** Cosa può controllare? (solitamente la spinta dei 4 motori).

4. SKRL: Il ponte tra RL e Robotica

- **skrl** è una libreria di Reinforcement Learning progettata specificamente per lavorare con ambienti vettorizzati come Isaac.



- **Integrazione:** skrl prenderà le "osservazioni" da Isaac Lab, le passerà alla tua rete (fatta con snnTorch) e restituirà le "azioni" al simulatore.
- **Il wrapper:** Dovrai usare un wrapper (già previsto in skrl) per rendere Isaac Lab compatibile con l'interfaccia degli agenti skrl.

5. La sfida tecnica: Integrare SNN in un loop di RL

Questa è la parte più complessa del tuo progetto. Le SNN sono intrinsecamente temporali, mentre molti algoritmi RL sono "frame-based".

Il flusso dei dati:

1. **Input:** Isaac Lab fornisce lo stato del drone (es. x, y, z).
2. **Encoding:** Devi convertire questi valori continui in spike (es. usando snnTorch.spikegen).
3. **SNN:** La rete elabora gli spike per T step temporali.

4. **Decoding:** Gli spike in uscita dalla rete vengono convertiti in un valore continuo (es. tensione dei motori).
5. **Azione:** Il drone si muove in Isaac Lab.

Roadmap consigliata per iniziare

1. **Setup Ambiente:** Installa Isaac Lab seguendo la documentazione ufficiale. È il passaggio più delicato (richiede driver NVIDIA aggiornati e Ubuntu 20.04/22.04).
2. **Test "Hello World" in Isaac:** Lancia l'esempio del Crazyflie fornito nel link di GitHub per vedere se il drone viene visualizzato correttamente.
3. **Tutorial snnTorch:** Segui i primi 3 tutorial sulla documentazione di snnTorch per capire come definire una rete e far passare i dati.
4. **Integrazione skrl:** Guarda gli esempi di skrl per "Isaac Orbit/Lab". Sostituisci la classica MlpPolicy (rete neurale standard) con una classe custom che eredita da nn.Module e contiene i neuroni di snnTorch.