LabActivity3 - Report

Analisi del problema

L’esercizio richiede l’implementazione di un robot in grado di rilevare la luce, raggiungerla nella maniera più veloce possibile, evitando ostacoli. Una volta arrivato in prossimità della luce, il robot dovrà fermarsi.

L’arena sarà formata dai soliti bordi di perimetro, da una fonte di luce e dal corrispettivo black spot che identificherà l’area dove fermarsi. Sono generati randomicamente un numero variabile di ostacoli e di robot.

Il problema dev’essere risolto implementando una **subsumption architecture**

Soluzione proposta

Ho diviso il comportamento del robot in 4 livelli gerarchici, ciascuno con una priorità crescente:

- **Livello 0 →** random walk. Priorità bassa. Il robot si muove in modo randomico aggiornando le velocità ogni MOVE_STEPS. Questo comportamento è eseguito solo se nessun altro comportamento più prioritario viene attivato.
- **Livello 1** → phototaxis. Priorità media. Viene effettuato un calcolo della media pesata angolare dell’intensità luminosa sui sensori circolari del robot, per stabilire la direzione della fonte di luce. Le ruote vengono regolate di conseguenza per convergere verso tale direzione. Se attivato, colora i LED di giallo
- **Livello 2** → avoid obstacle. Priorità alta. Il robot controlla i sensori frontali e laterali (1, 2, 23, 24) per evitare ostacoli. Se viene rilevata una distanza inferiore alla soglia OBSTACLE_THRESHOLD, il robot gira su sé stesso nel verso opposto all’ostacolo. I LED diventano blu.
- **Livello 3** → stop. Priorità massima. Attraverso il sensore *motor_ground*, il robot rileva superfici nere. Se rilevata, il robot si arresta completamente e colora i LED di rosso. Questa è l’unica condizione che modifica lo stato globale finish, interrompendo l'esecuzione dello step()

Ogni livello verifica se deve prendere il controllo del robot sulla base di soglie, impostando velocità e colore del LED. Utilizzo una variabile handler per evitare conflitti tra livelli inferiori e superiori.