

Nama : Angga Muhamad Gojali
NIM : 312210030
Kelas : TI.22.C.SE.1
Mata Kuliah : TIF701-Kecerdasan Buatan
Program Studi : Teknik Informatika
Dosen Pengampu : Yogi Yulianto, M.Kom

1. Pengertian Artificial Intelligence dan Contoh Penerapan

Menurut saya sendiri, Artificial Intelligence (AI) adalah cabang ilmu komputer yang berfokus pada pembuatan sistem yang mampu meniru cara berpikir, belajar, dan mengambil keputusan layaknya manusia. AI tidak sekadar menjalankan perintah, tetapi mampu beradaptasi dan memperbaiki kinerjanya berdasarkan data atau pengalaman sebelumnya.

Contoh penerapan AI dalam kehidupan sehari-hari:

- Asisten Virtual (seperti Google Assistant atau Siri): mampu memahami perintah suara, menjawab pertanyaan, dan membantu aktivitas pengguna sehari-hari.
- Sistem Rekomendasi (Netflix, YouTube, Tokopedia): menganalisis perilaku pengguna untuk memberikan saran konten atau produk yang sesuai dengan minat individu.

2. Pengertian Intelligent Agent

Intelligent Agent adalah entitas atau sistem yang mampu merasakan (perceive) lingkungannya melalui sensor dan mengambil tindakan (act) melalui aktuator berdasarkan persepsi tersebut agar mencapai tujuan tertentu. Agen cerdas biasanya dilengkapi dengan kemampuan untuk belajar dari pengalaman dan menyesuaikan strategi tindakannya ketika kondisi lingkungan berubah.

Contohnya bisa berupa robot pembersih lantai yang mendeteksi rintangan dan menyesuaikan arah agar dapat membersihkan ruangan secara efisien.

3. Analisis Satu Agent Cerdas dengan PEAS Framework dan Klasifikasi Lingkungan

Contoh Agent Cerdas: Robot vacuum cleaner (robot penyedot debu otomatis)

PEAS Framework:

- Performance Measure: kebersihan lantai, waktu pembersihan, efisiensi energi, jumlah area yang terlewati.
- Environment: ruangan rumah dengan berbagai bentuk dan rintangan seperti meja, kursi, atau dinding.
- Actuators: roda penggerak, penyedot debu, sikat pembersih.
- Sensors: sensor jarak, sensor debu, kamera (pada model tertentu), dan sensor posisi (gyroscope).

Klasifikasi Lingkungan:

- Observable: sebagian (karena tidak selalu tahu seluruh tata letak ruangan).
- Deterministic: sebagian tidak deterministik, karena hasil tindakan tergantung pada faktor acak seperti posisi rintangan.
- Static: dinamis (rintangan bisa berpindah seperti manusia berjalan).

- Discrete vs Continuous: kontinu, karena posisi dan pergerakan terjadi secara bertahap.
- Single vs Multi-agent: tunggal, karena robot bekerja sendiri.

4. Perbandingan Algoritma Blind Search (BFS, DFS, UCS)

NO	Aspek	Breadth-First Search (BFS)	Depth-First Search (DFS)	Uniform Cost Search (UCS)
1	Strategi	Menjelajahi semua node pada tingkat yang sama sebelum turun ke tingkat berikutnya.	Menjelajahi cabang tertentu sedalam mungkin sebelum berpindah ke cabang lain.	Memilih jalur dengan biaya paling rendah terlebih dahulu.
2	Penggunaan Memori	Tinggi, karena harus menyimpan semua node pada level saat ini.	Rendah, karena hanya menyimpan node di sepanjang jalur yang sedang dijelajahi.	Cukup tinggi, tergantung pada banyaknya jalur dan biaya yang dihitung.
3	Efektivitas (Optimalitas)	Optimal jika semua langkah memiliki biaya yang sama.	Tidak optimal, bisa berhenti di solusi yang tidak efisien.	Selalu optimal, karena memilih jalur dengan biaya terendah.

5. Perbedaan Global Search dan Local Search

Perbedaan Utama:

- Global Search mencari solusi dengan menjelajahi seluruh ruang kemungkinan secara sistematis untuk menemukan hasil terbaik secara menyeluruh.
- Local Search berfokus pada satu solusi awal, lalu memperbaikinya sedikit demi sedikit (iteratif) sampai menemukan solusi yang lebih baik di sekitar area tersebut.

Contoh:

- Global Search: Breadth-First Search (BFS) – menjelajahi seluruh node dari akar ke tingkat berikutnya tanpa melewati solusi potensial.
Cara kerja: memeriksa semua kemungkinan jalur mulai dari yang paling dangkal, lalu naik ke tingkat lebih dalam hingga menemukan solusi.
- Local Search: Hill Climbing Algorithm – memulai dari solusi acak dan bergerak menuju solusi tetangga yang memberikan peningkatan nilai.
Cara kerja: sistem terus berpindah ke posisi yang lebih “baik” hingga tidak ada perbaikan lebih lanjut (meski bisa terjebak di local optimum).