**Laporan Praktikum Minggu Ke-5 Kontrol Cerdas**

**Minggu ke-5**

Nama : Mahardhika Hendra Ahmad ramadana

NIM : 224308059

Kelas : TKA 7C

Akun Github (Tautan) : https://github.com/Mahardhikahendra

1. **Judul Percobaan**

Deep Refoircement Learning untuk control kompleks

1. **Tujuan Percobaan**

Tujuan dari percobaan pada praktikum kali ini, sebagai berikut:

1. Memahami konsep Deep Reinforcement Learning (DRL) dalam kontrol sistem kompleks.
2. Mengimplementasikan Deep Q-Network (DQN) untuk kontrol otomatis.
3. Menganalisis performa DRL dibandingkan dengan metode kontrol konvensional.
4. **Landasan Teori**

Deep Reinforcement Learning (DRL) adalah pengembangan dari Reinforcement Learning (RL) yang menggabungkan metode *deep learning* dengan prinsip *trial and error* pada RL. RL sendiri memungkinkan sebuah agen untuk belajar dari interaksi dengan lingkungannya melalui pemberian *reward* atau *punishment* sehingga agen dapat menemukan strategi terbaik secara mandiri. Namun, RL klasik menggunakan Q-Table untuk menyimpan nilai aksi pada setiap state, yang hanya efektif untuk masalah sederhana dengan ruang keadaan yang terbatas (Sutton and Barto, 2020).

Salah satu algoritma paling populer adalah Deep Q-Network (DQN), yang diperkenalkan oleh DeepMind dan berhasil mencapai performa tinggi dalam permainan Atari. DQN menggunakan dua teknik penting, yaitu *experience replay* untuk meningkatkan stabilitas pelatihan, serta strategi *epsilon-greedy* untuk menyeimbangkan eksplorasi dan eksploitasi dalam memilih aksi.

Deep Reinforcement Learning (DRL) dibandingkan dengan metode kontrol konvensional menjadi penting untuk menilai sejauh mana pendekatan ini mampu memberikan hasil yang lebih optimal, adaptif, dan efisien dalam pengendalian sistem dinamis. Berbeda dengan metode kontrol tradisional seperti PID atau LQR yang bergantung pada model matematis dan parameter tetap, DRL memiliki kemampuan untuk belajar langsung dari data interaksi serta menyesuaikan strategi pengendaliannya terhadap perubahan kondisi lingkungan. Dengan demikian, perbandingan ini diharapkan dapat memberikan gambaran yang lebih komprehensif mengenai keunggulan sekaligus tantangan penerapan DRL dalam sistem kontrol modern.

1. Apa kelebihan dan kelemahan DQN dibandingkan metode Q-Learning klasik?

DQN mampu belajar dari kondisi yang kompleks dan berdimensi tinggi, seperti gambar atau data sensor, tanpa perlu mendesain fitur secara manual. DQN juga dapat menemukan pola dan strategi optimal melalui pengalaman langsung, sehingga lebih fleksibel dan adaptif dibandingkan Q-Learning klasik., DQN juga memiliki kelemahan, antara lain DQN membutuhkan komputasi yang lebih besar dan waktu pelatihan lebih lama. Proses belajarnya menggunakan jaringan saraf yang bersifat black box, sehingga hasilnya sulit dianalisis dan dijelaskan secara langsung.

1. Bagaimana cara mengadaptasi DQN untuk kontrol sistem dinamis seperti drone atau robot industri?

Untuk mengadaptasi DQN pada drone atau robot industri, pertama-tama perlu ditentukan kondisi sistem (state) dan aksi yang bisa dilakukan (action). Fungsi reward dirancang agar agen belajar melakukan tugas dengan benar, misalnya menjaga kestabilan drone atau menempatkan lengan robot secara presisi. Jaringan saraf disesuaikan dengan jenis input, misalnya fully connected network untuk data sensor dan convolutional network untuk gambar. Agar proses belajar stabil, digunakan teknik seperti experience replay dan target network. Jika sistem memiliki aksi yang kontinu, DQN klasik bisa diganti dengan varian seperti DDPG. Pelatihan sebaiknya dilakukan di simulasi terlebih dahulu, lalu disesuaikan (fine tuning) sebelum diterapkan di dunia nyata agar aman dan lebih efektif..

1. **Assignment**

Pada praktikum ke-5 ini, dikembangkan sistem pengumpulan data gestur tangan secara real-time dengan memanfaatkan MediaPipe dan OpenCV.. Sistem ini berfungsi mendeteksi keberadaan tangan, menandai posisi setiap sendi (landmark), dan menyimpan koordinat titik-titik tersebut ke dalam file CSV sebagai dataset. Proses dimulai dengan inisialisasi modul mp.solutions.hands dari MediaPipe untuk mendeteksi tangan serta mp.solutions.drawing\_utils untuk menampilkan landmark pada video. Sebelum pelacakan dijalankan, program akan memeriksa apakah file *hand\_joints.csv* sudah tersedia. Jika belum, sistem otomatis membuat file baru dan menambahkan header berisi nomor frame, jenis tangan (kiri atau kanan), ID sendi (joint\_id), serta koordinat x, y, dan z.

Kamera diaktifkan menggunakan cv2.VideoCapture(0) untuk menangkap citra secara langsung. Setiap frame yang diterima dikonversi ke format RGB agar sesuai dengan input yang dibutuhkan oleh MediaPipe. Deteksi tangan dilakukan melalui model Hands() yang dapat mengenali hingga dua tangan dengan tingkat kepercayaan minimum 0.7 untuk deteksi maupun pelacakan. Jika tangan berhasil terdeteksi, sistem akan menggambar landmark dan koneksi antar sendi menggunakan fungsi mp\_drawing.draw\_landmarks(). Selain itu, setiap titik sendi diberi label nomor (0–20) dan dibungkus dengan bounding box berwarna hijau untuk memperjelas area tangan yang terdeteksi. Sistem juga menampilkan label “Right Hand” atau “Left Hand” di atas kotak tersebut berdasarkan hasil klasifikasi dari MediaPipe.

Semua hasil deteksi, termasuk koordinat x, y, dan z dari setiap titik sendi, secara otomatis disimpan dalam file CSV di setiap frame. Dataset ini nantinya dapat digunakan pada tahap berikutnya, misalnya untuk melatih model *machine learning* seperti SVM dalam mengenali gestur tangan. Program berjalan secara real-time hingga pengguna menekan tombol Esc, setelah itu kamera dan semua jendela ditutup. Dengan memadukan MediaPipe untuk ekstraksi fitur dan OpenCV untuk visualisasi, sistem ini dapat menghasilkan dataset koordinat tangan yang akurat serta terstruktur, sehingga menjadi dasar penting dalam pengembangan sistem pengenalan gestur tangan berbasis pembelajaran mesin.

1. **Data dan Output Hasil Pengamatan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Hasil Output** | **Deteksi** | **Posisi** | **Keterangan** |
| 1. | A screenshot of a computer  AI-generated content may be incorrect. | satu tangan berhasil terdeteksi yaitu tangan kanan | tangan kanan terbuka lebar dan Semua jari terlihat jelas | Garis hijau menghubungkan sendi, dan titik merah berada di setiap sendi |
| 2. |  | satu tangan berhasil terdeteksi yaitu tangan kiri | tangan kiri terbuka lebar dan Semua jari terlihat jelas | Garis berwarna hijau menghubungkan antar sendi, titik berwarna merah menandai posisi setiap sendi tangan |
| 3. |  | Dua tangan berhasil terdeteksi, yaitu tangan kanan dan tangan kiri. | Semua jari dari kedua tangan terbuka lebar dan terlihat jelas | Garis hijau menjadi penghubung sendi, dan titik merah berada di setiap sendi |

1. **Kesimpulan**

Sistem deteksi tangan dengan MediaPipe dan OpenCV berhasil mengenali tangan kanan dan kiri secara real-time dengan akurasi baik. Setiap tangan teridentifikasi melalui 21 titik sendi yang divisualisasikan dengan titik merah dan garis hijau, dan setiap titik terdapat penomoran yang divisualisasikan dengan warna putih, serta dibedakan otomatis antara “Left Hand” dan “Right Hand” menggunakan bounding box. Seluruh koordinat titik tangan tersimpan dalam file CSV dan dapat dimanfaatkan untuk melatih model pengenalan gestur dengan algoritma seperti Support Vector Machine (SVM). Secara keseluruhan, integrasi MediaPipe dan OpenCV terbukti efektif dan responsif dalam mendeteksi posisi serta bentuk gestur tangan manusia.

1. **Saran**

Sistem dapat ditingkatkan dengan menambah variasi gestur agar klasifikasi lebih adaptif, serta diuji pada berbagai pencahayaan dan sudut kamera untuk meningkatkan ketahanan deteksi.

1. **Daftar Pustaka**

Sutton, R.S., Barto, A., 2020. Reinforcement learning: an introduction, Second edition. ed, Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts London, England.