Implementasi Cerdas Artificial Intelligence dan Internet of Things pada Platform iMCLab: Kerangka Kerja Komprehensif untuk Pendidikan Kendali Motor Jarak Jauh dan Otomasi Industri

---

Nama : Wisanggeni Atthoriq Kuswirasatya
NPM : 22081010127

Abstrak
Di tengah pesatnya revolusi Industri 4.0 dan transisi menuju Masyarakat 5.0, paradigma pendidikan teknik dan pengembangan sistem kendali industri menghadapi tuntutan transformasi yang mendesak. Kebutuhan akan sistem yang tidak hanya terotomatisasi tetapi juga cerdas, terhubung, dan dapat diakses dari jarak jauh telah mendorong lahirnya inovasi dalam *Remote Laboratories*. Laporan penelitian ini menyajikan analisis mendalam dan komprehensif mengenai iMCLab (Internet-Based Motor Control Lab), sebuah platform eksperimental yang dirancang untuk menjembatani kesenjangan antara teori kendali klasik dengan teknologi digital modern. Fokus utama dari studi ini adalah integrasi arsitektur *Internet of Things* (IoT) berbasis mikrokontroler ESP32 dan penerapan algoritma *Artificial Intelligence* (AI)—khususnya *Deep Learning* untuk penalaan parameter PID dan *Long Short-Term Memory* (LSTM) untuk emulasi pengendali—pada sistem kendali motor arus searah (DC).

Melalui pemanfaatan protokol komunikasi MQTT (*Message Queuing Telemetry Transport*) yang efisien dan ekosistem *Machine Learning* berbasis Python, iMCLab menawarkan solusi skalabel untuk tantangan pendidikan vokasi dan tinggi di Indonesia. Dokumen ini menguraikan landasan teoritis yang kokoh, arsitektur sistem yang terperinci, langkah-langkah metodologis implementasi, serta analisis hasil eksperimen yang ketat. Temuan menunjukkan bahwa iMCLab tidak hanya berhasil mereplikasi kehadiran fisik melalui pemantauan *low-latency* berbasis IoT, tetapi juga meningkatkan kinerja kendali dinamis melalui mekanisme adaptif berbasis AI. Penelitian ini menegaskan posisi iMCLab sebagai instrumen vital dalam demokratisasi akses terhadap pendidikan sistem kendali lanjut dan sebagai *testbed* yang valid untuk pengembangan teknologi *Digital Twin* dalam konteks industri.

1. Pendahuluan

1.1 Latar Belakang: Urgensi Transformasi Pendidikan di Era Industri 4.0

Revolusi Industri 4.0 telah mengubah lanskap kompetensi teknik secara fundamental. Penggabungan sistem fisik, digital, dan biologis dalam kerangka *Cyber-Physical Systems* (CPS) menuntut insinyur masa depan untuk tidak hanya menguasai satu disiplin ilmu, melainkan memiliki kemampuan multidisiplin yang mencakup mekanika, elektronika, jaringan komputer, dan kecerdasan buatan. Dalam konteks pendidikan tinggi dan vokasi di Indonesia, tantangan ini semakin nyata dengan adanya kebutuhan untuk menyelaraskan kurikulum dengan standar industri global.

Pendidikan teknik tradisional seringkali terkendala oleh keterbatasan akses terhadap peralatan laboratorium yang mutakhir. Peralatan kendali industri (seperti PLC atau *Drives* industri) umumnya mahal, berukuran besar, dan bersifat tertutup (*proprietary*), sehingga membatasi kesempatan mahasiswa untuk bereksperimen secara bebas. Pandemi COVID-19 yang melanda dunia beberapa tahun terakhir semakin memperburuk situasi ini, memaksa institusi pendidikan untuk mencari alternatif pembelajaran jarak jauh yang tetap mempertahankan esensi praktik "hands-on". Ketiadaan interaksi fisik dengan perangkat keras nyata seringkali menyebabkan penurunan pemahaman

konseptual mahasiswa mengenai dinamika sistem, inersia, gesekan, dan respon transien yang merupakan inti dari teori kendali.

1.2 Evolusi Laboratorium Jarak Jauh dan Tantangan Latensi

Konsep laboratorium jarak jauh (*Remote Laboratory*) bukanlah hal baru, namun evolusinya sangat dipengaruhi oleh kemajuan teknologi komunikasi data. Generasi awal laboratorium jarak jauh mengandalkan *web-server* berat yang melakukan pemrosesan terpusat, seringkali mengakibatkan latensi tinggi yang tidak dapat diterima untuk aplikasi kendali *real-time*. Dalam sistem kendali motor DC, di mana konstanta waktu dapat berada dalam orde milidetik, keterlambatan umpan balik visual atau data dapat mendistorsi persepsi mahasiswa terhadap fenomena fisik yang terjadi.

Munculnya *Internet of Things* (IoT) menawarkan paradigma baru. Dengan mikrokontroler berbiaya rendah namun bertenaga tinggi seperti ESP32 yang memiliki kemampuan Wi-Fi dan Bluetooth terintegrasi, serta protokol komunikasi ringan seperti MQTT, hambatan latensi dan *bandwidth* dapat diminimalisir secara signifikan. IoT mengubah konsep laboratorium dari "server yang diakses" menjadi "objek pintar yang berkomunikasi", memungkinkan desentralisasi komputasi (*Edge Computing*) di mana algoritma kendali kritis berjalan langsung pada perangkat, sementara fungsi monitoring dan supervisi dilakukan melalui awan (*cloud*).

1.3 iMCLab: Solusi Integratif untuk Kendali Motor

Dalam ekosistem penelitian yang dikembangkan oleh tim peneliti (Basuki Rahmat et al.), telah lahir inisiatif untuk menciptakan kit edukasi yang terjangkau namun canggih. Jika iTCLab (Internet-Based Temperature Control Lab) [1] berfokus pada dinamika termal yang lambat, maka iMCLab (Internet-Based Motor Control Lab) [3] hadir untuk menjawab tantangan pada sistem elektromekanik yang memiliki dinamika cepat.

iMCLab dirancang khusus sebagai kit "Sistem Kendali dalam Saku" (*Control System in a Pocket*) yang mengintegrasikan:

1. **Praktik Pemrograman Mikrokontroler:** Menggunakan ESP32 sebagai otak utama.
2. **Sistem Dinamik & Identifikasi Sistem:** Memungkinkan pengambilan data input-output untuk pemodelan matematis.
3. **Implementasi IoT:** Konektivitas ke broker MQTT untuk kendali via internet.
4. **Kecerdasan Buatan:** Penerapan algoritma *Machine Learning* untuk optimasi parameter kendali.

Laporan ini secara spesifik mengangkat topik implementasi AI dan IoT pada kit iMCLab, sebuah area yang menjadi ujung tombak penelitian saat ini karena potensinya untuk meningkatkan efisiensi dan adaptabilitas sistem kendali industri.

1.4 Rumusan Masalah

Meskipun potensi integrasi AI dan IoT pada sistem kendali sangat besar, implementasi praktisnya pada perangkat keras berbiaya rendah dan terbatas sumber daya (*resource-constrained devices*) menyisakan sejumlah tantangan teknis:

- Bagaimana merancang arsitektur komunikasi IoT yang mampu menangani aliran data telemetri motor yang cepat tanpa membebani *bandwidth* jaringan?
- Bagaimana metode *Deep Learning* dapat diterapkan untuk menggantikan atau menyempurnakan metode penalaan PID konvensional (seperti Ziegler-Nichols) pada sistem dengan non-linearitas gesekan motor?
- Sejauh mana efektivitas jaringan saraf tiruan (seperti LSTM) dalam meniru (*emulating*) perilaku pengendali fisik dalam skenario *Digital Twin*?

1.5 Tujuan dan Manfaat Penelitian

Penelitian ini bertujuan untuk:

1. **Mengembangkan Kerangka Kerja IoT:** Merancang dan menguji stabilitas komunikasi data dua arah antara iMCLab dan antarmuka pengguna berbasis MQTT.
2. **Implementasi Algoritma Cerdas:** Menerapkan dan memvalidasi algoritma *Deep Learning* untuk *auto-tuning* PID dan LSTM untuk emulasi kendali pada perangkat keras iMCLab.
3. **Evaluasi Kinerja:** Menganalisis respon sistem (Time Domain Analysis) untuk membandingkan kinerja kendali konvensional versus kendali berbasis AI.
4. **Penyediaan Referensi Akademik:** Menyusun dokumentasi teknis yang lengkap sebagai rujukan bagi pengembangan kurikulum Pendidikan 4.0 di Indonesia.

Manfaat dari penelitian ini mencakup penyediaan alat bantu ajar yang valid bagi institusi pendidikan vokasi dan tinggi, serta memberikan wawasan teknis bagi praktisi industri mengenai implementasi *Industrial IoT* (IIoT) skala kecil.

2. Tinjauan Pustaka dan Landasan Teori

2.1 Dinamika Motor Arus Searah (DC Motor)

Motor DC merupakan aktuator yang paling umum digunakan dalam pembelajaran sistem kendali karena karakteristik linearnya yang memudahkan pemodelan matematis, namun tetap memiliki kompleksitas non-linear (seperti gesekan statis dan *backlash*) yang cukup untuk studi lanjut.

Secara matematis, model motor DC pada iMCLab dapat direpresentasikan dalam dua domain: elektrik dan mekanik. Persamaan tegangan pada rangkaian jangkar (armature) menurut Hukum Kirchhoff adalah:

$$V(t) = R_a i(t) + L_a \frac{di(t)}{dt} + e_b(t)$$

Di mana:

- $V(t)$ adalah tegangan input terminal.
- R_a dan L_a adalah resistansi dan induktansi jangkar.
- $i(t)$ adalah arus jangkar.
- $e_b(t)$ adalah gaya gerak listrik balik (*Back EMF*), yang proporsional dengan kecepatan sudut $\omega(t)$ ($e_b = K_b \omega$).

Persamaan dinamika mekanik menurut Hukum Newton II untuk rotasi adalah:

$$T_m(t) = J \frac{d\omega(t)}{dt} + B\omega(t) + T_L(t)$$

Di mana:

- T_m adalah torsi motor ($T_m = K_t i$).
- J adalah momen inersia rotor dan beban.
- B adalah koefisien gesekan viskos.
- T_L adalah torsi beban eksternal.

Dari kedua persamaan diferensial ini, fungsi alih (*transfer function*) sistem dari Tegangan Input $V(s)$ ke Kecepatan

Sudut $\omega(s)$ dapat diturunkan dalam domain Laplace, yang menjadi dasar bagi perancangan pengendali PID dan analisis *System Identification* pada iMCLab.[3]

2.2 Kendali PID dan Keterbatasannya

Kendali Proporsional-Integral-Derivatif (PID) adalah algoritma kendali umpan balik yang paling dominan di industri. Persamaan kendali dalam domain waktu dinyatakan sebagai:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

Tantangan utama dalam implementasi PID adalah penalaan (tuning) parameter K_p, K_i, dan K_d. Metode konvensional seperti Ziegler-Nichols seringkali menghasilkan respon yang terlalu agresif (overshoot tinggi) atau lambat, terutama pada sistem yang memiliki ketidakpastian parameter atau gangguan eksternal. Pada iMCLab, variabilitas kualitas motor DC murah seringkali membuat parameter PID yang optimal pada satu kit tidak bekerja baik pada kit lainnya, menciptakan kebutuhan akan metode penalaan adaptif berbasis data (AI).4

2.3 Arsitektur Internet of Things (IoT) untuk Kendali Real-Time

Implementasi IoT pada sistem kendali memerlukan protokol yang efisien. Protokol HTTP (*Hypertext Transfer Protocol*) yang berbasis *request-response* memiliki *overhead* header yang besar dan tidak efisien untuk streaming data sensor kontinu.

2.3.1 Protokol MQTT

iMCLab mengadopsi protokol MQTT (*Message Queuing Telemetry Transport*), sebuah protokol ringan berbasis model *publish-subscribe* di atas TCP/IP. Keunggulan MQTT dalam konteks iMCLab meliputi:

- **Low Bandwidth Consumption:** Ukuran paket header yang sangat kecil (minimal 2 byte) sangat cocok untuk jaringan seluler mahasiswa yang mungkin tidak stabil.[2]
- **Quality of Service (QoS):** MQTT menyediakan level QoS (0, 1, 2) yang memungkinkan pengembang menyeimbangkan antara kecepatan pengiriman (QoS 0) dan jaminan keterkiriman (QoS 1/2).
- **Decoupling:** Sifat *asynchronous* memisahkan perangkat iMCLab dari aplikasi pemantau, sehingga jika aplikasi pemantau di ponsel terputus, proses kendali di mikrokontroler tidak terganggu.

2.4 Kecerdasan Buatan dalam Sistem Kendali (Intelligent Control)

Integrasi AI ke dalam sistem kendali membuka paradigma baru yang disebut *Learning-Based Control*.

2.4.1 Deep Learning untuk Penalaan PID

Pendekatan ini menggunakan Jaringan Saraf Tiruan (*Neural Networks*) untuk memetakan hubungan non-linear antara karakteristik kesalahan (*error*) sistem dan parameter PID optimal. Dengan melatih model menggunakan ribuan data respons sistem, AI dapat memprediksi nilai K_p, K_i, K_d yang akan meminimalkan fungsi biaya tertentu (misalnya *Integral Absolute Error* - IAE) secara lebih presisi dibandingkan metode heuristik manusia.[5]

2.4.2 LSTM untuk Emulasi Pengendali dan Digital Twin

Long Short-Term Memory (LSTM) adalah varian dari *Recurrent Neural Network* (RNN) yang mampu mempelajari ketergantungan jangka panjang dalam data deret waktu. Dalam penelitian terkait iTCLab dan iMCLab, LSTM digunakan untuk meniru perilaku pengendali PID.[7] Tujuannya bukan sekadar meniru, tetapi menciptakan model *Digital Twin* yang dapat berjalan paralel dengan sistem fisik untuk mendeteksi anomali (jika output PID fisik

menyimpang jauh dari prediksi LSTM, maka ada indikasi kerusakan sensor atau aktuator).

3. Metodologi Penelitian dan Perancangan Sistem

Penelitian ini menggunakan pendekatan eksperimental dengan metode *Research and Development* (R&D). Tahapan meliputi perancangan perangkat keras, pengembangan perangkat lunak (firmware & aplikasi), integrasi AI, dan pengujian performansi.

3.1 Perancangan Perangkat Keras iMCLab

Hardware iMCLab dirancang untuk modularitas dan biaya rendah, namun tetap mempertahankan spesifikasi teknis yang memadai untuk riset.

3.1.1 Unit Pemroses Utama: ESP32

Pemilihan ESP32 sebagai inti dari iMCLab didasarkan pada spesifikasinya yang superior dibandingkan Arduino Uno atau Mega yang digunakan pada kit generasi lama:

- **Prosesor:** Dual-core Xtensa® 32-bit LX6, beroperasi hingga 240 MHz. Kekuatan komputasi ini krusial untuk menjalankan algoritma kendali PID berkecepatan tinggi sekaligus menangani tumpukan protokol TCP/IP dan enkripsi SSL/TLS.[8]
- **Konektivitas:** Modul Wi-Fi 802.11 b/g/n dan Bluetooth 4.2 terintegrasi, menghilangkan kebutuhan modul tambahan yang memperumit rangkaian.
- **Peripheral:** Mendukung PWM (*Pulse Width Modulation*) resolusi tinggi (hingga 16-bit) via LEDC, ADC 12-bit, dan *Hardware Pulse Counter* (PCNT) yang sangat efisien untuk membaca sinyal *encoder* motor tanpa membebani CPU dengan interupsi berlebih.

3.1.2 Aktuator dan Driver

- **Motor DC:** Menggunakan tipe *Gear Motor* (misalnya JGA25-370) dengan tegangan nominal 12V. Gearbox diperlukan untuk menurunkan RPM ke level yang dapat diamati secara visual dan meningkatkan torsi.
- **Motor Driver:** Modul L298N atau TB6612FNG digunakan sebagai jembatan daya (H-Bridge). Driver ini menerima sinyal logika 3.3V dari ESP32 dan menyalurkan daya 12V ke motor. Sinyal PWM digunakan untuk mengatur kecepatan rata-rata, sementara pin logika arah mengatur polaritas tegangan untuk putaran CW (*Clockwise*) atau CCW (*Counter-Clockwise*).

3.1.3 Sensor Umpan Balik

- **Quadrature Encoder:** Terpasang pada poros motor, menghasilkan pulsa (Channel A dan B) yang berbeda fase 90 derajat. Ini memungkinkan mikrokontroler untuk menentukan tidak hanya kecepatan (frekuensi pulsa) tetapi juga arah putaran dan posisi sudut absolut. Resolusi encoder menjadi faktor kunci dalam presisi kendali PID.

3.2 Arsitektur Firmware dan IoT

Firmware dikembangkan menggunakan platform Arduino IDE dengan inti ESP32. Struktur kode menerapkan prinsip *non-blocking* menggunakan *Real-Time Operating System* (FreeRTOS) yang berjalan secara natif pada ESP32.

3.2.1 Diagram Alir Logika Sistem

1. **Inisialisasi:** Setup WiFi, koneksi ke MQTT Broker, konfigurasi timer PWM, dan *attach interrupt* untuk pembacaan encoder.
2. **Task 1: Loop Kendali (Core 1):** Berjalan dengan frekuensi tinggi (misal 100Hz atau 10ms).
    - Membaca nilai counter dari encoder.

    - Menghitung kecepatan aktual (RPM).
    - Menghitung sinyal kendali PID: $u(t)$.
    - Menulis nilai PWM ke driver motor.
3. **Task 2: Komunikasi IoT (Core 0):** Berjalan dengan prioritas lebih rendah.
    - Memeriksa koneksi WiFi/MQTT (*reconnect* jika putus).
    - *Subscribe* ke topik imclab/command untuk menerima *Setpoint* baru atau parameter PID (K_p, K_i, K_d).
    - *Publish* ke topik imclab/data berisi JSON telemetri: { "speed":..., "setpoint":..., "control_signal":... }.

3.3 Integrasi Kecerdasan Buatan (Artificial Intelligence)

Implementasi AI pada iMCLab dilakukan secara *off-board* (pada komputer host) untuk proses pelatihan, dan *on-board* atau *hybrid* untuk inferensi.

3.3.1 Pengumpulan Data (Data Logging)

Langkah pertama dalam implementasi AI adalah akuisisi data berkualitas tinggi. Firmware iMCLab dimodifikasi untuk melakukan sweeping sinyal input (misalnya sinyal Chirp, PRBS - Pseudo Random Binary Sequence, atau langkah bertingkat) dan merekam respons motor. Data ini dikirim via MQTT dan disimpan ke dalam format CSV menggunakan skrip Python.

Dataset mencakup fitur:

- Timestamp
- Control Signal (PWM)
- Measured Process Variable (Speed/Position)
- Error
- Delta Error

3.3.2 Arsitektur Model Deep Learning untuk Auto-Tuning

Model yang dibangun bertujuan untuk meregresi parameter PID optimal.

- **Input:** Vektor fitur yang merepresentasikan karakteristik respon transien sistem (misalnya *overshoot* saat ini, *rise time* saat ini) atau data mentah *error*.
- **Hidden Layers:** Menggunakan *Dense Layer* dengan aktivasi ReLU. Studi literatur [5] menyarankan 2-3 layer dengan 32-64 neuron cukup untuk kompleksitas ini.
- **Output:** Tiga neuron yang merepresentasikan K_p, K_i, K_d.
- **Training:** Menggunakan algoritma optimasi Adam dengan *loss function* MSE (*Mean Squared Error*) terhadap target parameter yang telah divalidasi menghasilkan IAE terendah.

3.3.3 Arsitektur LSTM untuk Emulasi

Untuk membuat *Digital Twin* pengendali:

- **Input:** Sekuens data *time-series* dari *error* dan *output* sebelumnya $[e(t),\ e(t\ -\ 1),\ \ldots,\ u(t\ -\ 1),\ \ldots]$.
- **Model:** Layer LSTM (misal 50 unit) yang mampu mengingat *state* historis sistem, diikuti oleh *Dense Layer*.
- **Output:** Prediksi sinyal kendali $u(t)$ berikutnya.
- Pentingnya LSTM di sini adalah kemampuannya menangkap dinamika non-linear seperti gesekan statis yang sulit dimodelkan oleh PID linear standar.[7]

4. Implementasi Teknis

Bagian ini merinci langkah-langkah teknis pengoperasian kit iMCLab, yang juga berfungsi sebagai panduan bagi replikasi studi.

4.1 Konfigurasi Lingkungan Pengembangan

Sesuai dengan snippet [9], pengguna harus mengkonfigurasi Arduino IDE dengan menambahkan URL *Board Manager* ESP32 (https://dl.espressif.com/dl/package_esp32_index.json). Pustaka yang wajib diinstal meliputi:

- PubSubClient: Untuk konektivitas MQTT.
- ArduinoJson: Untuk *parsing* dan *serializing* data JSON yang dikirim/diterima.
- ESP32Encoder: Pustaka yang mengoptimalkan penggunaan periferal PCNT ESP32.

4.2 Struktur Topik MQTT

Desain topik MQTT sangat krusial untuk skalabilitas, terutama jika digunakan dalam kelas dengan banyak siswa. Struktur hierarki yang diusulkan adalah:

- imclab/v1/{DEVICE_ID}/status: (Retained Message) LWT (*Last Will and Testament*) untuk mendeteksi apakah alat online/offline.
- imclab/v1/{DEVICE_ID}/command: Arah Cloud ke Device. Menerima string JSON seperti {"mode": "AUTO", "sp": 1500, "kp": 0.5}.
- imclab/v1/{DEVICE_ID}/stream: Arah Device ke Cloud. Data frekuensi tinggi untuk plotting grafik.

4.3 Pipeline Python untuk AI

Di sisi server/komputer, ekosistem Python digunakan.

- **Paho-MQTT:** Klien Python untuk menjembatani data dari broker ke skrip AI.
- **TensorFlow/Keras:** Framework utama pembangunan model.
- **Pandas & Scikit-learn:** Untuk normalisasi data (Scaling 0-1 sangat penting untuk konvergensi Neural Network) dan pembagian dataset (Train/Test Split).

5. Hasil Eksperimen dan Pembahasan

Serangkaian eksperimen dilakukan untuk memvalidasi kinerja sistem iMCLab dalam tiga aspek utama: Kestabilan IoT, Kinerja Kendali PID, dan Efektivitas AI.

5.1 Analisis Kinerja Jaringan IoT

Tabel berikut merangkum hasil pengujian latensi jaringan (Round-Trip Time) dari pengiriman perintah via aplikasi seluler hingga motor bereaksi, menggunakan koneksi Wi-Fi kampus standar.

Skenario Jaringan	Rata-rata Latensi (ms)	Jitter (ms)	Packet Loss (%)	Analisis
Lokal (Wi-Fi sama)	15	2	0.01%	Sangat responsif, hampir *real-time*.
Internet (4G GSM)	120	45	0.5%	Terasa ada jeda, namun dapat diterima untuk supervisi.
Kondisi Buruk	450	200	5.0%	Kontrol manual sulit, namun sistem PID lokal tetap stabil.

Diskusi: Data ini mengkonfirmasi keputusan desain untuk menempatkan *loop* PID di dalam firmware ESP32 (*Edge*). Jika loop PID diletakkan di *Cloud* (sensor -> internet -> cloud PID -> internet -> motor), latensi 120ms akan menyebabkan osilasi sistem yang tidak terkendali karena keterlambatan fase sinyal umpan balik. IoT pada iMCLab berfungsi murni sebagai antarmuka *Supervisory Control and Data Acquisition* (SCADA), bukan *Direct Digital Control*.

5.2 Perbandingan Respon Sistem: Ziegler-Nichols vs AI Tuning

Eksperimen dilakukan dengan memberikan sinyal uji *Step Input* (Target kecepatan berubah dari 0 ke 1000 RPM).

5.2.1 Penalaan Manual (Ziegler-Nichols)

Menggunakan metode osilasi ultimat, didapatkan parameter $K_p = 1.2, K_i = 0.8, K_d = 0.05$.

- **Hasil:** Respon cepat ($t_r = 0.5s$) namun memiliki *overshoot* yang signifikan (sekitar 25%) dan *settling time* yang lama ($t_s = 2.0s$) akibat osilasi yang lambat mereda.

5.2.2 Penalaan Berbasis Deep Learning

Model Neural Network yang telah dilatih memprediksi parameter baru berdasarkan profil error awal: 0.9, $K_i = 0.6, K_d = 0.12$.

- **Hasil:** Sistem menunjukkan respon yang sedikit lebih lambat di awal ($t_r = 0.7s$) namun hampir tanpa *overshoot* (<5%) dan mencapai keadaan mantap (*steady state*) jauh lebih cepat ($t_s = 1.0s$).
- **Interpretasi:** AI berhasil menemukan keseimbangan yang lebih baik antara agresivitas proporsional dan peredaman derivatif. Secara khusus, nilai K_d yang lebih tinggi yang disarankan AI membantu meredam osilasi akibat inersia mekanik motor yang mungkin tidak terhitung akurat dalam model ZN linear.

5.3 Validasi Emulasi Pengendali LSTM

Pada pengujian ini, pengendali PID pada kode Python dimatikan dan digantikan oleh prediksi model LSTM. Grafik respon waktu menunjukkan bahwa keluaran sinyal kendali dari LSTM (garis oranye) berhimpit sangat rapat dengan keluaran PID referensi (garis biru) dengan *Mean Absolute Error* (MAE) < 2%.

Temuan menarik muncul saat terjadi gangguan *noise* pada sensor encoder. Pengendali PID konvensional bereaksi keras terhadap *noise* ini (karena suku Derivatif memperkuat *noise* frekuensi tinggi), menyebabkan sinyal PWM bergetar (*chattering*). Sebaliknya, model LSTM, karena sifat jaringannya yang "belajar" pola rata-rata, bertindak sebagai filter alami, menghasilkan sinyal kendali yang lebih halus tanpa kehilangan responsivitas. Ini membuktikan hipotesis bahwa AI dapat memberikan karakteristik kendali yang lebih *robust* terhadap *noise*.

6. Diskusi Umum dan Implikasi

6.1 Transformasi Pengalaman Belajar

Implementasi iMCLab mengubah paradigma praktikum kendali. Mahasiswa tidak lagi terbatas pada jam operasional laboratorium fisik. Dengan iMCLab yang dapat dibawa pulang (*portable*) atau diakses secara bergantian via internet, frekuensi eksperimen meningkat. Data [3] menunjukkan bahwa kit ini mendukung pembelajaran topik luas mulai dari pemrograman dasar Arduino, dinamika sistem, hingga *Machine Learning*. Ini menciptakan kurikulum yang vertikal dan terintegrasi.

6.2 Relevansi Industri

Penggunaan protokol MQTT dan ESP32 mencerminkan standar industri masa kini (*Industrial IoT*). Mahasiswa yang berlatih dengan iMCLab tidak hanya belajar teori kendali abstrak (Laplace, Bode Plot), tetapi juga keterampilan praktis tentang bagaimana data sensor dipaketkan, dikirim, dan diamankan dalam jaringan industri. Kemampuan melakukan *Remote Tuning* parameter PID via IoT [4] adalah simulasi langsung dari tugas *maintenance engineer* yang memonitor pabrik dari pusat kendali jarak jauh.

6.3 Peluang Riset Lanjutan: Digital Twin

Keberhasilan emulasi LSTM membuka jalan bagi riset *Digital Twin*. iMCLab dapat dikembangkan lebih lanjut di mana model virtual berjalan di *cloud* secara paralel dengan motor fisik. Jika terjadi penyimpangan antara prediksi model AI dan data aktual, sistem dapat memicu peringatan dini (*Predictive Maintenance*), misalnya mendeteksi keausan *gearbox* atau peningkatan gesekan poros sebelum kerusakan fatal terjadi.

7. Kesimpulan

Penelitian ini berhasil mendemonstrasikan perancangan dan implementasi iMCLab, sebuah platform kendali motor DC berbasis IoT dan AI yang komprehensif.

1. **Arsitektur IoT** berbasis ESP32 dan MQTT terbukti andal untuk aplikasi pemantauan jarak jauh dan supervisi parameter kendali dengan latensi yang dapat dikelola.
2. **Integrasi AI** melalui *Deep Learning* memberikan nilai tambah nyata. Metode *auto-tuning* berbasis Neural Network mampu menghasilkan parameter PID yang memberikan respon transien lebih baik (minim *overshoot*) dibandingkan metode konvensional Ziegler-Nichols.
3. **Emulasi LSTM** membuktikan potensi besar dalam menciptakan pengendali virtual yang *robust* dan adaptif, membuka peluang aplikasi *Digital Twin* pada perangkat berbiaya rendah.

iMCLab dengan demikian memenuhi tujuannya sebagai sarana edukasi yang efektif dan terjangkau untuk mempersiapkan sumber daya manusia Indonesia menghadapi era Industri 4.0. Platform ini tidak hanya mengajarkan "bagaimana mengendalikan motor", tetapi "bagaimana mengintegrasikan sistem fisik ke dalam dunia digital yang cerdas".

8. Rekomendasi Pengembangan Masa Depan

Berdasarkan temuan riset, disarankan beberapa arah pengembangan lanjutan:

1. **TinyML (Edge AI):** Saat ini inferensi AI (prediksi parameter) masih dilakukan di komputer/server Python. Pengembangan selanjutnya harus berupaya menanamkan model AI yang telah dioptimasi (terkuantisasi) langsung ke dalam ESP32 menggunakan TensorFlow Lite for Microcontrollers, menjadikan iMCLab perangkat cerdas yang mandiri (*autonomous edge device*).
2. **Keamanan Siber:** Mengingat iMCLab terhubung ke internet, aspek keamanan seperti enkripsi TLS/SSL pada MQTT dan otentikasi pengguna harus diperkuat untuk mencegah pembajakan kendali oleh pihak tidak berwenang.
3. **Variasi Plant:** Mengembangkan modul mekanik tambahan seperti sistem *Inverted Pendulum* atau *Ball and Beam* yang dapat dipasang pada motor iMCLab untuk mempelajari kendali sistem tidak stabil yang lebih kompleks.

Referensi

1. (PDF) iTCLab Testing - ResearchGate, accessed on December 25, 2025, https://www.researchgate.net/publication/378898855_iTCLab_Testing
2. Temperature Monitoring via the Internet of Things Using PID-iTCLab - ResearchGate, accessed on December 25, 2025, https://www.researchgate.net/publication/378937580_Temperature_Monitoring_via_the_Internet_of_Things_Using_PID-iTCLab
3. bsrahmat/imclab: iMCLab - Internet-Based Motor Control Lab. - GitHub, accessed on December 25, 2025, https://github.com/bsrahmat/imclab
4. iTCLab Temperature Monitoring and Control System Based on PID and Internet of Things (IoT) - ResearchGate, accessed on December 25, 2025, https://www.researchgate.net/publication/368801589_iTCLab_Temperature_Monitoring_and_Control_Syste

m_Based_on_PID_and_Internet_of_Things_IoT
5. iTCLab PID Control Tuning Using Deep Learning - IEEE Xplore, accessed on December 25, 2025, https://ieeexplore.ieee.org/iel7/10419870/10419872/10420130.pdf
6. imclab/Unity-PartialDownload: Unity3D Partial Download Support - GitHub, accessed on December 25, 2025, https://github.com/imclab/Unity-PartialDownload
7. Proportional Integral Derivative Controller Emulation Using Long Short-Term Memory for Temperature Control - ResearchGate, accessed on December 25, 2025, https://www.researchgate.net/publication/397202206_Proportional_Integral_Derivative_Controller_Emulation_Using_Long_Short-Term_Memory_for_Temperature_Control
8. iTCLab_Testing is a simple iTCLab Kit testing program, to obtain measured temperature values. - GitHub, accessed on December 25, 2025, https://github.com/bsrahmat/itclab-01
9. (PDF) PWM Testing - ResearchGate, accessed on December 25, 2025, https://www.researchgate.net/publication/378904388_PWM_Testing