BAB II
LANDASAN TEORI

2.1. Tinjauan Pustaka

Dalam penelitian ini, akan digunakan referensi atau tinjauan pustaka berdasarkan latar belakang yang telah diuraikan sebelumnya. Beberapa tinjauan pustaka dari penelitian terdahulu dan beberapa penelitian yang relevan dengan penelitian yang akan dilakukan oleh penulis akan digunakan untuk mendukung penulisan skripsi. Informasi lebih lanjut tentang hal ini dapat ditemukan pada Tabel 2.1 yang disajikan di bawah ini.

Tabel 2.1 Tinjauan Pustaka

No.	Judul Penelitian	Tahun	Hasil
1.	Penelitian oleh Natanael dan Wahab (2020) dalam jurnalnya yang berjudul "Perancangan Pengontrol *Adaptive Fuzzy PID* pada *Brushless* DC *Motor*"	2020	Penelitian ini mengevaluasi penerapan pengontrol *Adaptive Fuzzy PID (AFPID)* pada *brushless* DC motor (BLDC) untuk meningkatkan performa *respons transien* dibandingkan dengan pengontrol PID konvensional. Hasil simulasi menunjukkan bahwa AFPID memberikan performa yang lebih baik pada parameter *rise time* (0.364s), *overshoot* (4.737%), dan *settling time* (2.971s). Temuan ini menunjukkan potensi *AFPID* sebagai pengontrol yang efektif dalam mengoptimalkan performa BLDCM (Natanael & Wahab, 2020).
2.	Penelitian oleh Wibowo dkk. (2021) dalam jurnalnya yang berjudul	2021	Penelitian ini berfokus pada pengembangan kontrol keseimbangan untuk robot *hexapod*

	"Kontrol Keseimbangan Robot *Hexapod* EILERO menggunakan *Fuzzy Logic*"		EILERO. Penggunaan *fuzzy logic* sebagai sistem kontrol keseimbangan menggunakan umpan balik data kemiringan dari sensor *IMU*. Hasil pengujian menunjukkan bahwa robot dapat menyeimbangkan diri pada kemiringan antara -15° hingga 15° pada orientasi *roll* dan *pitch*. Robot merespons dengan stabil dalam waktu di bawah 3000ms, membuatnya lebih stabil saat bergerak di permukaan tidak datar (Wibowo dkk., 2021).
3.	Penelitian oleh Saputro dkk. (2023) dalam jurnalnya yang berjudul "Implementasi Sistem Kendali Keseimbangan Statis Pada Robot *Quadruped* Menggunakan *Reinforcement Learning*"	2023	Penelitian ini mengatasi masalah keseimbangan pada robot *quadruped* dengan menerapkan metode *reinforcement learning* menggunakan algoritma *Q-Learning*. Hasil pembelajaran pada simulator Gazebo menunjukkan bahwa sistem dapat berjalan dengan baik dan robot *quadruped* dapat menyeimbangkan diri pada sumbu *roll* dan *pitch*. Kelebihan dari pendekatan ini adalah kemampuan untuk melakukan pembelajaran tanpa perhitungan matematis yang rumit. Namun, kekurangannya adalah proses pembelajaran memerlukan waktu dan penggunaan simulator, sehingga

			perlu diuji lebih lanjut dalam situasi nyata. Pengembangan lebih lanjut dapat memberikan manfaat bagi pekerjaan manusia dengan robot yang dapat berfungsi dalam berbagai kondisi keseimbangan (Saputro dkk., 2023).
4.	Penelitian oleh Asrofi dkk. (2015) dalam jurnalnya yang berjudul "Stabilisasi Robot Berkaki 6 *(Hexapod)* Pada Bidang Miring Menggunakan 9 *DOF IMU* Berbasis *Invers Kinematic*"	2015	Penelitian ini berfokus pada perancangan dan pembuatan sistem stabilisasi untuk robot *hexapod* pada permukaan miring menggunakan sensor 9 *DOF IMU* berbasis *invers kinematik*. Sensor IMU terdiri dari *accelerometer*, *gyroscope*, dan *magnetometer*, yang memberikan masukan sudut kemiringan dan *heading* robot. Kendali *fuzzy-PID* digunakan untuk menjaga *body* robot tetap datar pada permukaan miring. Pengujian sensor IMU menunjukkan nilai RMSE total sumbu *pitch* sebesar 1,73%, *roll* sebesar 1,67%, dan *yaw* sebesar 1,24%. Pada pengujian keseluruhan, sistem menunjukkan respon yang paling stabil dengan konstanta deffuzifikasi Kp *(proportional)* bernilai 0,5, 1, dan 3, Ki *(integral)* bernilai 0,5 untuk semua parameter, dan Kd *(derivative)* bernilai 0,25, 0,35, dan 0,45. Dengan implementasi ini,

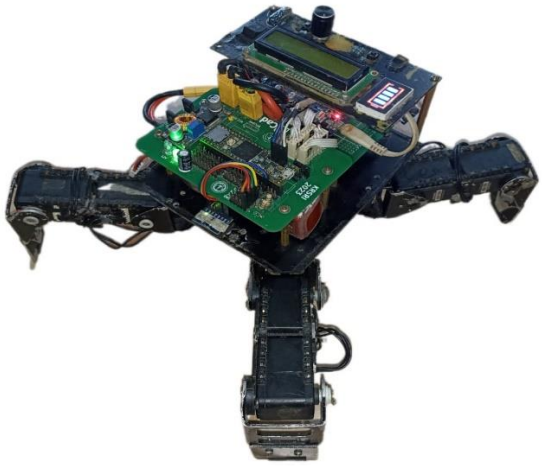
			robot *hexapod* dapat bergerak secara *translasi* sesuai dengan perubahan sumbu x, y, dan z pada *invers kinematik* (Asrofi dkk., 2015).
5.	Penelitian oleh Li dkk. (2019) dalam jurnalnya yang berjudul "Upright balance control of hexapod robot based on quadratic error differential PID algorithm"	2019	Penelitian ini bertujuan untuk mengatasi masalah kontrol keseimbangan tegak pada robot *hexapod*. Sebuah algoritma kontrol *PID* baru diusulkan, berdasarkan metode persamaan *Lagrange*, untuk memodelkan keadaan keseimbangan tegak robot *hexapod*. Untuk masalah kontrol keseimbangan tegak pada robot *hexapod*, diusulkan pengaturan parameter *quadratic error differential* dengan modul *secant saturation*. Hasil simulasi mengkonfirmasi bahwa algoritma yang diusulkan memungkinkan dan mampu mencapai kinerja kontrol keseimbangan yang lebih baik dibandingkan dengan metode tradisional (Li dkk., 2019).

2.2. Robot Berkaki

Robot berkaki adalah jenis robot yang dirancang dan dibuat dengan sistem atau struktur yang mirip dengan kaki manusia atau hewan berkaki. Robot berkaki mampu bermanuver dengan kaki-kaki buatan, baik dengan 2 kaki yang sering disebut dengan robot *humanoid*, berkaki tiga (*tripod*), berkaki empat (*quadraped*), robot berkaki enam(*hexapod*) dan robot berkaki banyak lainnya (Kurniawan, 2022; Sherrod, 2019; Zhang dkk., 2021).

Jenis robot berkaki yang akan di gunakan pada penelitian ini adalah *quadruped*, yang artinya robot tersebut memiliki empat kaki. *Quadruped* robot meniru struktur kaki hewan empat kaki seperti anjing, kucing, kuda, atau hewan lainnya yang berjalan dengan menggunakan empat titik kontak pada permukaan (Zhang dkk., 2021).

Robot ini dirancang untuk bergerak dan berjalan menggunakan algoritme kontrol yang rumit, yang bisa menjadi sulit untuk dirancang dan dapat mengakibatkan biaya produksi yang tinggi (Sherrod, 2019). Robot berkaki empat dapat digunakan untuk berbagai tugas, seperti pencarian dan penyelamatan di lingkungan yang tidak terstruktur. Bentuk robot *quadruped* dapat dilihat pada Gambar 2.1.

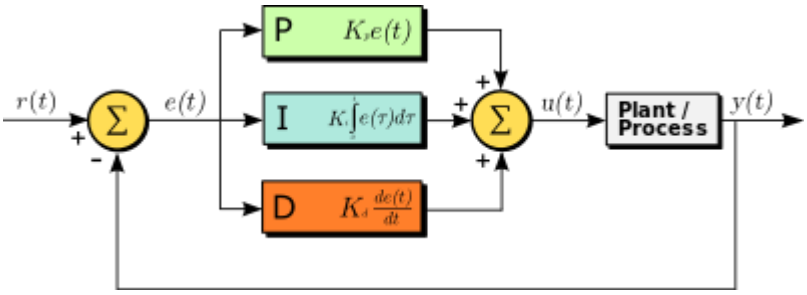


Gambar 2.1 Robot Quadruped

2.3. Kendali *PID*

PID (*Proportional-Integral-Derivative*) adalah sebuah mekanisme yang menentukan akurasi dari sistem instrumentasi dengan menggunakan umpan balik. Komponen kontroler yang ada dalam sistem memiliki dampak yang signifikan terhadap perilaku sistem, dan karakteristik *plant* harus dapat diterima agar perubahan perilaku sistem hanya dimungkinkan melalui tambahan sub-sistem, yaitu kontroler (Jayadi dkk., 2022). Pengontrol PID adalah jenis sistem kontrol yang menggunakan fungsi proporsional, integral, dan turunan untuk mengatur proses. Aksi proporsional memberikan output yang sebanding dengan kesalahan antara setpoint dan variabel proses, aksi integral memberikan output yang sebanding dengan integral kesalahan dari waktu ke waktu, dan aksi turunan

memberikan output yang sebanding dengan laju perubahan kesalahan. Kombinasi dari ketiga tindakan ini memungkinkan pengontrol untuk merespons perubahan dalam variabel proses dan mempertahankan setpoint (Merrikh-Bayat dkk., 2015; Oziablo dkk., 2020).



Gambar 2.2 Diagram blok kendali PID
(Jayadi dkk., 2022)

Telah ditemukan secara empiris bahwa apa yang disebut pengontrol PID adalah struktur yang berguna. Karakteristik dari algoritma PID dapat digambarkan sebagai berikut:

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)\,d\tau + T_d\frac{de(t)}{dt}\right) \cdots\cdots\cdots\cdots\cdots\cdots (2.1)$$

di mana u adalah variabel kontrol dan e adalah kesalahan kontrol ($e = y_{sp} - y$). Variabel kontrol merupakan hasil dari tiga suku: P (yang berkaitan dengan kesalahan proporsional), I (yang berkaitan dengan integral dari kesalahan), dan D (yang berkaitan dengan turunan dari kesalahan). Parameter kontrol PID adalah gain proporsional K, waktu integral Ti, dan waktu derivatif Td (Åström dkk., 1995).

2.3.1 Aksi Proposional

Dalam kasus kontrol proporsional murni, persamaan kendali dari Persamaan (2.2) berkurang menjadi

$$u(t) = K_e(t) + u_b \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2.2)$$

Aksi kontrol hanya sebanding dengan *error control*. Variabel u_b adalah bias atau reset. Ketika kesalahan kontrol $e$adalah nol, variabel kendali mengambil nilai $u(t) = u_b$. Bias u_b sering kali ditetapkan ke $(u_{max} + u_{min})/2$, tetapi kadang-kadang dapat disesuaikan secara manual sehingga kesalahan kontrol statis adalah nol pada *setpoint* yang diberikan (Åström dkk., 1995).

2.3.2 Aksi Integral

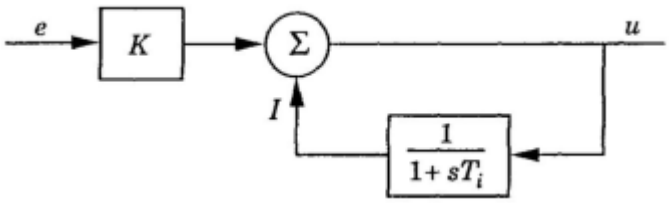
Fungsi utama dari aksi integral adalah untuk memastikan bahwa output proses sesuai dengan setpoint dalam kondisi steady state. Dengan kontrol proporsional, biasanya ada kesalahan kontrol dalam kondisi steady state. Dengan Dengan aksi integral, kesalahan positif yang kecil akan selalu mengarah pada sinyal kontrol yang meningkat, dan kesalahan negatif akan memberikan sinyal kontrol yang menurun tidak peduli seberapa kecil kesalahannya.

Argumen sederhana berikut menunjukkan bahwa kesalahan kondisi *steady state* akan selalu nol dengan aksi integral. Asumsikan bahwa sistem berada dalam kondisi *steady state* dengan sinyal kontrol konstan (u_0) dan kesalahan konstan(e_0). Dari Persamaan (2.3) dapat disimpulkan bahwa sinyal kontrol integral adalah sebagai berikut:

$$u_0 = K\left(e_0 + \frac{e_0}{T_i}t\right) \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(2.3)$$

Selama $e_0 \neq 0$, hal ini jelas bertentangan dengan asumsi bahwa sinyal kendali u_0 adalah konstan. Pengontrol dengan aksi integral akan selalu memberikan kesalahan kondisi tunak nol.

Aksi integral juga dapat divisualisasikan sebagai perangkat yang secara otomatis mengatur ulang istilah bias u_0, dari pengontrol proporsional. Hal ini diilustrasikan dalam diagram blok pada Gambar 2.3, yang menunjukkan kontroler proporsional dengan reset yang disesuaikan secara otomatis. Penyesuaian Penyesuaian dilakukan dengan mengumpan balik sinyal, yang merupakan nilai keluaran yang difilter, ke titik penjumlahan pengontrol. Ini sebenarnya adalah salah satu penemuan awal dari aksi integral, atau "reset otomatis".



Gambar 2.3 Diagram blok kendali Integral

2.3.3 Aksi *Derivative*

Tujuan dari tindakan derivatif adalah untuk meningkatkan stabilitas loop tertutup. Karena dinamika proses, perlu beberapa waktu sebelum perubahan dalam variabel kontrol terlihat dalam output proses.

Dengan demikian, sistem kontrol akan terlambat dalam mengoreksi kesalahan. Aksi kontroler dengan aksi proporsional dan derivatif dapat diartikan sebagai kontrol yang dibuat proporsional dengan output proses yang diprediksi, dimana prediksi dibuat dengan mengekstrapolasi kesalahan dengan garis singgung pada kurva kesalahan. Struktur dasar dari kontroler PD adalah

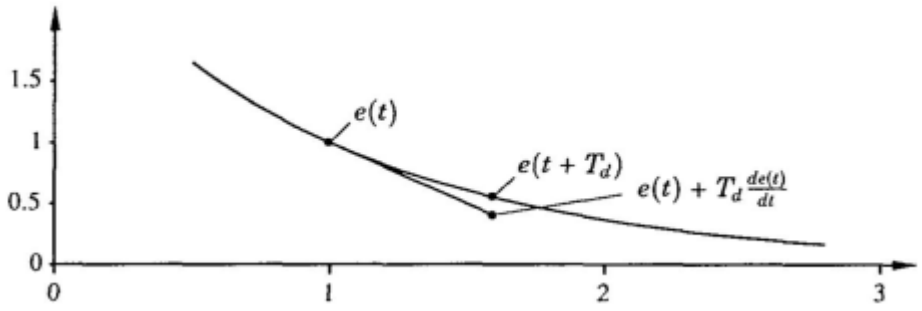
$$u(t) = K\left(e(t) + T_d \frac{de(t)}{dt}\right) \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2.4)$$

Pengembangan deret Taylor dari $e(t + T_d)$ memberikan

$$e(t + T_d) \approx e(t) + T_d \frac{de(t)}{dt} \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2.5)$$

Sinyal kontrol dengan demikian sebanding dengan estimasi control error pada waktu T_d ke depan, di mana estimasi diperoleh dengan ekstrapolasi linier.

Sifat-sifat aksi turunan diilustrasikan pada Gambar 2.4, yang menunjukkan simulasi sistem dengan kontrol PID. Kontroler gain dan waktu integrasi dijaga konstan, $K = 3$ dan $T_i = 2$, dan waktu turunan Td diubah. Untuk Td = 0, kita memiliki kontrol PI murni. Sistem loop tertutup berosilasi dengan parameter yang dipilih. Awalnya redaman meningkat dengan bertambahnya waktu turunan, tetapi menurun lagi ketika waktu turunan menjadi terlalu besar.



Gambar 2.4 Grafik kendali Derivative

2.4. Logika *Fuzzy*

Logika *fuzzy* merupakan salah satu metode untuk menentukan keputusan berbasis sebab dan akibat (if-then). *Fuzzy* didefinisikan sebagai samar-samar atau kabur, dengan derajat keanggotaan yang dimiliki *fuzzy* adalah 0 dan 1. *Fuzzy logic* pertama kali dikenalkan kepada publik oleh Zadeh, seorang profesor di University of California di Berkeley. Teori dan Konsep *Fuzzy Logic* dipresentasikan bukan sebagai suatu metodologi kontrol sampai tahun 1970, tetapi sebagai suatu cara pemrosesan data dengan memperkenankan penggunaan partial set membership dibanding crisp set membership atau non-membership. Pada tahun 1972, Profesor Zadeh memperkenalkan Logika *Fuzzy* sebagai rasional fungsi kendali.

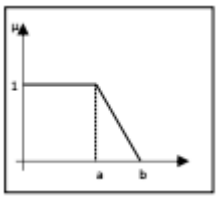
Logika *fuzzy* adalah suatu cara yang tepat untuk memetakan suatu ruang input dalam suatu ruang output. Pada logika *fuzzy* terdapat tiga metode yaitu, metode Tsukamoto, metode Mamdani, dan metode Sugeno. Penentuan model *inference* harus tepat sesuai data yang ingin diolah karena perlakuan setiap metode pada *fuzzy inference* sistem berbeda dan mempunyai karakteristik fungsi keanggotaan dan himpunan *fuzzy* (Alimuddin, 2020). Secara matematika hal ini dinyatakan pada persamaan (2.6).

$$\mu A : U \rightarrow [0,1] \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2.6)$$

Beberapa pendekatan dalam perancangan aturan *fuzzy* telah dirancang untuk kondisi yang terdiri dari tiga jenis fungsi keanggotaan, yaitu *reversegrade*, segitiga, dan *grade* (Siregar, 2018).

a) Fungsi keanggotaan *reverse grade* dapat dilihat pada Gambar 2.5 dengan persamaan matematisnya ditunjukkan pada Persamaan 2.7.

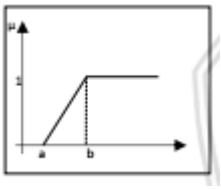
$$\mu[x] = \begin{cases} 0, x \geq b \\ \frac{x-a}{b-a}, a < x < b \\ 1, x \leq a \end{cases} \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2.7)$$



Gambar 2.5 Grafik Fungsi Keanggotaan reverse grade
(Siregar, 2018)

b) Fungsi keanggotaan *grade* dapat dilihat pada Gambar 2.6 dengan persamaan matematisnya ditunjukkan pada Persamaan 2.8.

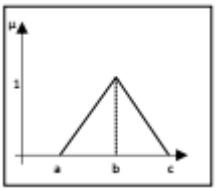
$$\mu[x] = \begin{cases} 0, x \leq b \\ \frac{-x+a}{b-a}, a < x < b \\ 1, x \geq b \end{cases} \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2.8)$$



Gambar 2.6 Grafik Fungsi Keanggotaan grade
(Siregar, 2018)

c) Sedangkan fungsi keanggotaan segitiga dapat dilihat pada Gambar 2.7 dengan persamaan matematisnya ditunjukkan pada Persamaan 2.9.

$$\mu[x] = \begin{cases} 0, x \geq c \; atau \; x \leq a \\ \frac{x-a}{b-a}, a < x < b \\ \frac{-x+c}{b-a}, b < x < c \\ 1, x = b \end{cases} \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2.9)$$



Gambar 2.7 Grafik Fungsi Keanggotaan segitiga
(Siregar, 2018)

2.4.1 Fuzzifikasi

Fuzzifikasi merupakan proses mengubah variabel non-fuzzy (variabel numerik) menjadi variabel *fuzzy* (variabel linguistik). Variabel numerik adalah variabel yang memiliki himpunan nilai berupa bilangan angka. Sementara itu, variabel linguistik adalah istilah untuk menyebutkan kelompok yang mewakili

suatu keadaan atau kondisi tertentu menggunakan bahasa, seperti "muda", "paruh baya", dan "tua".

Pada awal sistem, fungsi keanggotaan harus ditentukan, di mana himpunan *fuzzy* telah ditetapkan sebelumnya. Himpunan *fuzzy* merupakan sekelompok nilai yang mewakili suatu kondisi dalam variabel *fuzzy* (Alimuddin, 2020).

2.4.2 Rule Base Fuzzy

Fuzzy Rule dalam pengendalian berbasis fuzzy adalah bagian yang menggambarkan dinamika suatu sistem terhadap masukan yang dikarakteristikan oleh sekumpulan variabel-variabel linguistik dan berbasis pengetahuan seorang operator ahli. Penyataan ini biasanya diwujudkan dalam bentuk pernyataan bersyarat "IF_THEN". Sistem Multi Input Single Output (MISO) menggunakan aturan pengendalian fuzzy berbentuk sebagai berikut:

1. Rule 1: IF X adalah A1 DAN Y adalah B1 THEN Z adalah C1
2. Rule 2: IF X adalah A2 DAN Y adalah B2 THEN Z adalah C2
3. ...
4. Rule n: IF X adalah An DAN Y adalah Bn THEN Z adalah Cn

Dalam bentuk aturan tersebut, X dan Y adalah variabel masukan yang mewakili berbagai nilai linguistik, sedangkan Z adalah variabel keluaran sistem yang juga memiliki nilai linguistik. An, Bn, dan Cn adalah nilai-nilai linguistik yang merepresentasikan karakteristik dari masing-masing variabel X, Y, dan Z (Siregar, 2018).

2.4.3 Defuzzifikasi

Defuzzifikasi adalah tahap penting dalam sistem pengendalian berbasis *fuzzy* di mana himpunan *fuzzy* yang dihasilkan dari komposisi aturan-aturan *fuzzy* diubah menjadi nilai crisp dalam domain himpunan *fuzzy* tersebut. Tujuan dari proses ini adalah untuk memperoleh nilai konkret dan mudah dimengerti sebagai hasil keluaran. Terdapat berbagai metode defuzzifikasi yang dapat digunakan, seperti AI (*adaptive integration*), BADD (*basic defuzzification distributions*), BOA (*bisector of area*), CDD (*constraint decision defuzzification*), COA (*center of area*), COG

(*center of gravity*), LOM *(last of maximum*), MeOM (*mean of maxima*), dan MOM (*middle of maximum*). Dari sekian banyak metode tersebut, metode COG atau centroid adalah salah satu yang sering dipilih karena memberikan hasil yang intuitif dan mewakili dengan mengambil titik pusat pada daerah *fuzzy*. Dengan demikian, defuzzifikasi berperan sebagai jembatan penting antara konsep *fuzzy* dan implementasi nyata, memungkinkan sistem pengendalian berbasis *fuzzy* untuk memberikan solusi yang lebih adaptif dan tepat dalam menghadapi situasi kompleks.

2.5. Adaptive Fuzzy PID

Prinsip Adaptasi Implisit Pengontrol PID melibatkan representasi pengontrol PID sebagai dua pengontrol yang terhubung secara paralel: pengontrol PID utama dan pengontrol PID tambahan. Pengontrol PID utama ditentukan oleh parameter "*kernel*", sedangkan pengontrol PID tambahan dibedakan oleh parameter adaptif. Adaptasi parameter adaptif mengarah pada adaptasi implisit parameter PID.

Implementasi kendali AFPID melibatkan pengembangan hukum adaptasi untuk mendapatkan sinyal K, T_i, dan T_d. Fuzzy Logic Controller (FLC) digunakan untuk menghasilkan sinyal K, T_i, dan T_d. FLC memiliki dua input: kesalahan sistem $e(t)$ dan perubahannya $ec(t)$. FLC memiliki tiga keluaran: K, T_i, dan T_d.

Fungsi keanggotaan untuk *input* dan *output* FLC didefinisikan menggunakan himpunan *fuzzy*. Penyetelan K, T_i, dan T_d didasarkan pada tiga aturan praktis yang berkaitan dengan besarnya kesalahan sistem dan perubahannya. Strategi kontrol pada AFPIDC yang diusulkan didasarkan pada aturan fuzzy yang menentukan nilai K, T_i, dan T_d berdasarkan input $e(t)$ dan $ec(t)$ (Yuanhui Yang dkk., 2010).

Pendekatan kendali AFPID yang diusulkan memungkinkan adaptasi implisit dari parameter PID tanpa secara langsung membuatnya adaptif. Pendekatan ini bertujuan untuk menghasilkan serangkaian sinyal yang secara tidak langsung mengadaptasi parameter PID.

2.6. Teensy 3.6

Teensy 3.6 adalah sebuah papan pengembangan dengan mikrokontroler ARM Cortex-M4 32-bit dengan kecepatan 180 MHz. *Board* ini memiliki banyak fitur dan

fasilitas, termasuk dukungan untuk banyak protokol komunikasi, memori internal 1 MB Flash dan 256 KB RAM, *port input/output* digital I/O, PWM, dan analog *input*, serta dukungan untuk koneksi *Ethernet* dan *Wi-Fi*. Teensy 3.6 cocok untuk pengembangan berbagai jenis aplikasi seperti robotika, kontrol motor, dan kontrol cahaya.

Pada penelitian ini Teensy 3.6 akan digunakan sebagai pengendali navigasi robot dengan menggunakan data yang diambil dari RPLIDAR, memproses data dari RPLIDAR dan mengontrol gerakan robot, bentuk fisik Teensy 3.6 dapat dilihat pada gambar 2.1.

Berikut adalah tabel spesifikasi dari Teensy 3.6 yang dapat dilihat pada tabel 2.2 di bawah ini :

Tabel 2.2 Spesifikasi Teensy 3.6

Prosessor	Cortex-M4F
Kecepatan *Clock*	180 Mhz
Tegangan Operasional	3,3V
Flash Memori	1024 Kbytes
RAM	256 Kbytes
EEPROM	4096 bytes
Digital I/O	58 buah
Analog *Input*	25 buah
PWM Output	22 buah
USB Port	2 buah
Port Komunikasi	6 buah Serial, 3 buah SPI, 4 buah I2C, 2 CAN Bus

(Sumber : www.pjrc.com)

Gambar 2.8 Teensy 3.6
(Sumber : www.pjrc.com)

2.7. OpenCM 9.04

OpenCM 9.04 adalah sebuah papan pengembangan berbasis mikrokontroler ARM Cortex-M3 32-bit dengan kecepatan 72 MHz. Board ini memiliki fitur dan fasilitas khusus, termasuk dukungan untuk protokol komunikasi yang digunakan

untuk mengendalikan servo Dynamixel AX-18 yang menggunakan protokol komunikasi DYNAMIXEL Protocol 1.0 serta sambungan fisik TTL Level Multi Drop Bus.

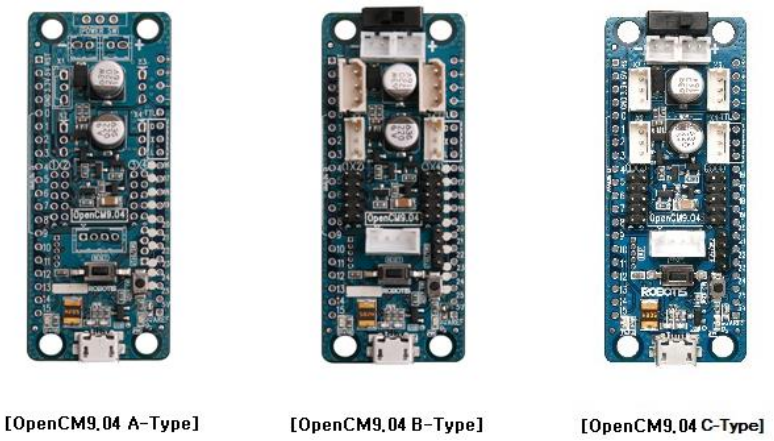
Pada penelitian ini OpenCM 9.04 akan digunakan sebagai subsistem yang memiliki fungsi kinematika untuk mengendalikan kaki-kai robot dengan menggunakan firmware Metabot dari Rhoban bentuk fisik OpenCM 9.04 dapat dilihat pada gambar 2.2.

Berikut adalah tabel spesifikasi dari OpenCM 9.04 yang dapat dilihat pada tabel 2.3 di bawah ini :

Tabel 2.3 Spesifikasi OpenCM 9.04

Processor	STM32F103CB (ARM Cortex-M3)
Tegangan Operasi	5V ~ 16V
Pin I/O	GPIO x 26
Timer	4 (16bit)
Analog Input(ADC)	10 (12bit)
Flash	128Kb
SRAM	20Kb
Clock	72Mhz
Port Komunikasi	1xUSB, 3xUART, 2x I2C (TWI), 2xSPI
Debug	JTAG & SWD
DYNAMIXEL TTL BUS	4 (Max 1Mbps)

(Sumber : www.emanual.robotis.com)



Gambar 2.9 OpenCM 9.04
(Sumber : www.emanual.robotis.com)

2.8. AX-18A

Dynamixel AX-18 merupakan sebuah komponen yang berbentuk *servo* yang diproduksi oleh perusahaan ROBOTIS. *Servo* ini telah mengintegrasikan sepenuhnya motor DC, mikrokontroler ATMEGA 8, *driver* motor, dan jaringan lainnya dalam satu modul, menjadikannya sebagai tipe *smart aktuator*. Selain itu, *servo* ini telah dilengkapi dengan berbagai sensor termasuk sensor posisi yang berupa variabel resistor untuk mengenali posisi dan arah putaran *servo* saat beroperasi, sensor suhu, sensor beban, dan sensor tegangan. Sebagian besar material gigi (*gear*) pada *servo* AX-18 terbuat dari bahan plastik dengan kombinasi gigi logam, menghasilkan torsi yang lebih besar dibandingkan dengan seri-seri yang lebih rendah. *Servo* ini mampu berkomunikasi dengan kendali utama melalui komunikasi UART TTL *half duplex*, dengan pemasangan elektronik antar *servo* yang menggunakan *daisy chain*. Bentuk fisik Dynamixel AX-18 digambarkan dalam Gambar 2.12.

Gambar 2.10 *Servo* AX-18A
(Sumber : www.emanual.robotis.com)

2.9. Modul MPU 6050

MPU-60X0 adalah perangkat *MotionTracking* 6-axis terintegrasi yang menggabungkan giroskop 3-axis, akselerometer 3-axis, dan *Digital Motion Processor™* (*DMP*). Dengan bus sensor I2C khusus, perangkat ini secara langsung menerima input dari kompas 3-axis eksternal untuk memberikan *output MotionFusion™* 9-axis yang lengkap. Perangkat MotionTracking MPU-6050, dengan integrasi 6-axis, *MotionFusion ™ on-board*, dan *firmware* kalibrasi *run-time*, memungkinkan pengguna untuk menghilangkan pemilihan, kualifikasi, dan integrasi tingkat sistem yang mahal dan rumit dari perangkat diskrit, menjamin kinerja gerak yang optimal bagi konsumen. MPU-6050 juga dirancang untuk

berinteraksi dengan beberapa sensor digital noninersia, seperti sensor tekanan, pada port I2C tambahan.

MPU-6050 memiliki tiga konverter analog-ke-digital (ADC) 16-bit untuk mendigitalkan *output* giroskop dan tiga ADC 16-bit untuk mendigitalkan *output* akselerometer. Untuk pelacakan presisi baik cepat maupun lambat gerakan, bagian-bagiannya memiliki rentang skala penuh giroskop yang dapat diprogram pengguna dari ± 250, ± 500, ± 1000, dan ± 2000 ° / detik (dps) dan rentang skala penuh akselerometer yang dapat diprogram pengguna dari ± 2g, ± 4g, ± 8g, dan ± 16g (*MPU-6000 and MPU-6050*, 2013).

Gambar 2.11 Modul MPU 6050