BAB 2
DASAR TEORI

2.1 KAJIAN PUSTAKA

Dalam penelitian ini, referensi yang diambil dari berbagai penelitian terdahulu yang berkaitan dengan topik penelitian dan dapat dijadikan acuan penelitian yang bertujuan untuk menetapkan batasan penelitian lebih lanjut. Dalam kajian pustaka ini, terdapat beberapa hasil penelitian yang dijadikan referensi dalam melakukan penelitian.

Penelitian pertama yang bertujuan mengembangkan alat deteksi gas CO menggunakan IoT dan memperjelas hasil pengujian alat pemantauan gas CO. Sistem menggunakan sensor MQ-7, Arduino Nano, dan modul *WiFi* ESP32. Perbandingan antara sensor MQ-7 dan CO Meter menunjukkan selisih rata-rata 4,70 ppm untuk gas buang kendaraan dan 3,79 ppm untuk asap rokok. Penelitian ini memungkinkan pemantauan jarak jauh dan pengiriman data CO secara *real-time,* namun hanya dapat memonitor kadar gas CO tanpa kontrol sirkulasi udara di kabin mobil [7].

Penelitian kedua dengan menggunakan metode eksperimental dengan melakukan uji coba alat ketika AC mati dan AC hidup. Implementasi sistem mengggunakan sensor MQ-7, sensor suhu DHT-22, Arduino UNO dan modul *WiFi* ESP32. Dari penelitian tersebut menghasilkan alat monitoring gas CO dan suhu pada kabin mobil dengan memanfaatkan IoT untuk pengiriman peringatan/notifikasi melalui telegram. Pada penelitian tersebut tidak membahas mengenai bagaimana apabila tidak ada *wireles/router* pada daerah tersebut dan hanya membahas mengenai monitoring gas CO dan suhu dengan menggunakan notifikasi telegram yang dapat mengirimkan data kadar gas CO dan suhu terkini kepada pengendara [8].

Kemudian penelitian ketiga bertujuan merancang alat pendeteksi dan pemantauan gas CO menggunakan aplikasi *Visual Basic*. Sistem ini menggunakan sensor MQ-7, modul *Bluetooth* HC-05, dan Arduino UNO. Kadar gas CO

ditampilkan melalui layar LCD, dan *buzzer* akan berbunyi jika gas CO terdeteksi. Berdasarkan pengujian, lampu hijau menyala saat kondisi aman, lampu kuning saat kondisi waspada, dan lampu merah serta *buzzer* menyala saat kondisi bahaya. Alat ini mampu memonitoring gas CO di udara melalui layar LCD dan *software Visual Basic* [9].

Kemudian penelitian keempat bertujuan untuk mengetahui kadar gas CO, HC dan CO2 pada gas buang kendaraan bermotor yang dilakukan pada 15 kendaraan berbeda. Pada penelitian ini digunakan Arduino UNO, Sensor MQ-7 untuk mengukur gas CO, Sensor MG-811 untuk mengukur gas HC, sensor TGS-2201 untuk mengukur gas CO2 dan LCD 16x2. Penelitian tersebut dilakukan dengan membandingkan atau mengkalibrasi beberapa sensor yang digunakan dengan menggunakan alat *Techniotest Infrared Multi Gas*. Berdasarkan penelitian ini, didapatkan nilai *error relative* pada gas CO sebesar 10,17%, gas HC sebesar 8,5% dan gas CO2 sebesar 1,6%. Hasil dari 15 percobaan pada kendaraan, setiap pengiriman melalui *sms gateway* menggunakan *visual basic* 6.0 rata-rata membutuhkan waktu *delay* sebesar 17 detik [10].

Kemudian penelitian kelima mengembangkan penelitian sebelumnya dengan menambahkan kontrol power window yang otomatis terbuka jika sensor mendeteksi gas CO melebihi ambang batas. Sistem menggunakan mikrokontroler NodeMCU ESP8266, sensor MQ-7, buzzer, dan *power window*. Alat ini memonitor dan mendeteksi gas CO di kabin mobil, membuka jendela otomatis dan menyalakan *buzzer* saat kadar CO melebihi 35 ppm. Implementasi IoT memungkinkan mikrokontroler mengirim data sensor ke aplikasi *Firebase* melalui internet [11].

Selanjutnya penelitian keenam bertujuan merancang sistem pemantauan gas CO dan CO2 yang ditampilkan melalui layar LCD dan mengirim informasi ke aplikasi Telegram menggunakan internet. Sistem menggunakan mikrokontroler ESP32, dua sensor MQ-7 untuk mendeteksi CO, dan sensor MQ-135 untuk mendeteksi CO2. Setelah kalibrasi, sensor MQ-7 memiliki error sebesar 0,87% dan sensor MQ-135 sebesar 0,66%. Sistem ini mencapai akurasi 99% dalam pemantauan gas CO dan CO2 [12].

Berdasarkan beberapa teknologi yang berkembang saat ini yaitu tentang alat pendeteksi gas CO, terdapat teknologi yang dapat dikembangkan salah satunya

yaitu alat yang berfungsi untuk mendeteksi gas CO dalam kabin mobil. Dengan menggunakan sensor MQ-7 yang dapat mendeteksi gas CO yang mana terdapat indikator LED, *Buzzer* dan LCD sebagai *output*. Selain itu, menggunakan pengendali *exhaust fan* untuk membuang gas CO yang ada pada kabin mobil agar udara di dalam kabin mobil dapat bersirkulasi.

2.2 DASAR TEORI

2.2.1 Pencemaran Udara

Pencemaran udara atau pencemaran atmosfer mengacu pada adanya zat-zat fisik, biologi dan kimia pada atmosfer dalam jumlah yang berbahaya bagi kesehatan manusia dan organisme hidup lainnya. Pencemaran udara dibedakan menjadi pencemaran primer, yang berasal langsung dari sumbernya seperti gas CO yang dihasilkan dari pembakaran, dan polusi sekunder yang terbentuk dari reaksi polutan primer di atmosfer, seperti ozon fotokimia. Penyebabnya bisa berasal dari alam atau aktivitas manusia. Udara segar terdiri dari 78% nitrogen, 21% oksigen, dan 1% gas lain, dengan ciri tidak berbau, segar, sejuk, dan ringan dihirup. Polusi udara dapat menyebabkan kesulitan bernapas dan berbagai kerusakan lingkungan, berdampak negatif terutama pada manusia [13].

Tabel 2.1 Sumber dan Standar Kesehatan Emisi Gas Buang [13]

Pencemar	Sumber	Keterangan
Karbon Monoksida (CO)	Asap kendaraan bermotor dan industry	Standar Kesehatan: 10 mg/m^3 (10 ppm)
Sulfurdioksida (SO_2)	Panas dan pembangkit listrik	Standar Kesehatan: 80 $\mu g/m^3$ (0.03 ppm)
Particullate Matter	Asap kendaraan bermotor dan proses industry	Standar Kesehatan: 50 $\mu g/m^3$ selama 1 tahun
Nitrogen Dioksida (NO_2)	Asap kendaraan bermotor, panas dan fasilitas	Standar Kesehatan: 100 pg/m^3 (0.05 ppm) selama 1 jam
Ozon (O_3)	Atmosfer	Standar Kesehatan: 235 $\mu g/m^3$ (0.12 ppm) selama 1 jam

Tabel 2.1 merupakan sumber dan standar kesehatan emisi gas buang. Gas CO dihasilkan dari asap kendaraan bermotor atau industri dan standar kesehatan pada gas CO yaitu 10 mg/m^3 atau setara dengan 10 ppm. Kemudian gas SO_2

bersumber dari panas atau dari pembangkit listrik dan standar kesehatan pada gas SO_2 yaitu 80 µg/m^3 atau setara dengan 0.03 ppm. Gas *particullate matter* berasal dari asap kendaraan bermotor atau pada industri dan standar kesehatannya yaitu 50 µg/m^3 selama 1 tahun. Selanjutnya pada gas NO_2 berasal dari asap kendaraan bermotor atau dari panas dan standar kesehatan dari gas NO_2 yaitu 100 pg/m^3 atau setara dengan 0.05 ppm selama 1 jam. Kemudian gas O_3 bersumber dari atmosfer dan standar kesehatannya yaitu 235 µg/m^3 yang setara dengan 0.12 ppm selama 1 jam.

2.2.2 Karbon Monoksida (CO)

Gas CO merupakan suatu gas yang tidak memiliki bau, tidak memiliki rasa, dan tidak berwarna. Gas CO dihasilkan dalam bentuk gas buang yang sangat beracun ketika bahan bakar fosil tidak terbakar sempurna dengan udara. Di kota besar dengan lalu lintas yang padat memiliki konsentrasi CO di atmosfer yang lebih tinggi dibandingkan daerah pedesaan, dengan kendaraan bermotor sebagai sumber utama polutan CO (sekitar 59,2%). Kendaraan berbahan bakar bensin menghasilkan lebih banyak CO dibandingkan kendaraan berbahan bakar solar. Konsentrasi CO di udara dipengaruhi oleh laju emisi dan pembuangan CO dari udara [13].

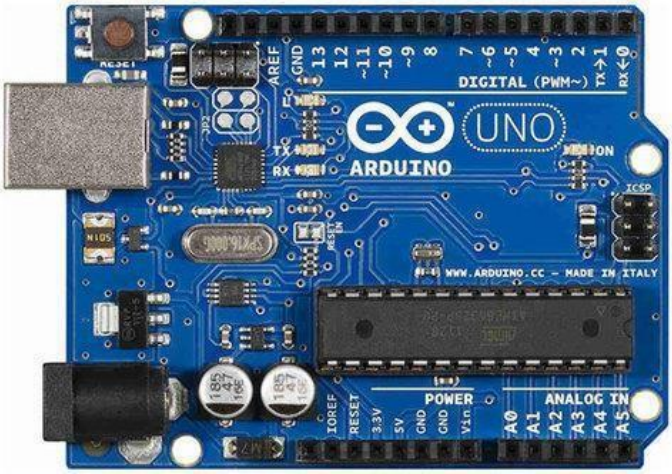
Gas CO sangat reaktif dengan hemoglobin darah, dan afinitas hemoglobin (Hb) dibandingkan afinitas Hb terhadap O2. Pada saat gas CO dihirup melalui saluran hidung dan masuk ke dalam darah, maka gas CO lebih cepat berikatan dengan Hb dibandingkan oksigen. Sehingga dapat mengurangi kadar oksigen dalam tubuh dan menyebabkan pusing, sakit kepala, dan dapat menyebabkan keracunan [13].

2.2.3 *Hardware*

2.2.3.1 Arduino UNO

Arduino UNO adalah salah satu jenis papan mikrokontroler yang populer dalam proyek elektronik dan pemrograman. Papan ini berbasis mikrokontroler ATmega328 dan banyak digunakan oleh pemula dan ahli dalam bidang elektronika untuk berbagai macam proyek, mulai dari proyek sederhana hingga kompleks.

Arduino Uno dapat diberi daya melalui koneksi USB atau dari catu daya. Jika ingin menggunakan sumber listrik, bisa menggunakan adaptor DC atau baterai. Adaptor dapat dihubungkan ke soket adaptor pada port *supply*. Mikrokontroler yang digunakan pada Arduino menggunakan jenis *Dual Inline Package* (DIP) [14].



Gambar 2.1 Arduino Uno R3 [14]

Gambar 2.1 merupakan board Arduino Uno R3. Arduino Uno memiliki 14 pin *input* dan *output* digital dimana 6 pin *input* tersebut dapat digunakan sebagai *output* PWM dan 6 pin *input* analog, 16 MHz osilator kristal, koneksi USB, *jack power*, ICSP *header*, dan tombol reset. Untuk mendukung mikrokontroler agar dapat digunakan, cukup hanya menghubungkan *board* Arduino Uno ke komputer dengan menggunakan kabel USB atau listrik dengan AC ke adaptor DC atau baterai untuk menjalankannya. Setiap 14 pin digital pada arduino uno dapat digunakan sebagai *input* dan *output*, menggunakan fungsi *pinMode*(), *digitalwrite*(), dan *digitalRead*() [14].

Tabel 2.2 Spesifikasi Arduino Uno R3 [15]

Mikrokontroler	Atmega328
Power Supply	5 Volt
Input Supply	7-12 Volt
Jumlah Pin I/O Digital	14
Jumlah Pin Analog	6
Arus DC Tiap Pin I/O	40 mA
Memori Flash	32 KB

Tabel 2.2 merupakan spesifikasi dari Arduino Uno R3. Arduino uno menggunakan jenis mikrokontroller Atmega328. Tegangan pengoperasian yang digunakan pada Arduino sebesar 5*volt* DC. Pengoperasian Arduino Uno disarankan menggunakan tegangan input sebesar 7-12 *volt*. Arduino Uno memiliki jumlah pin *input* dan *output* digital berjumlah 14 yang dapat digunakan sebagai *input* atau *output*, menggunakan fungsi *pinMode*(), *digitalWrite*(), dan *digitalRead*(). Jumlah pin analog pada Arduino Uno berjumlah 6 pin. Arus DC pada setiap pin *input* atau *output* Arduino Uno sebesar 40 mA. Arduino memiliki 32 KB *flash memory* yang berfungsi untuk menyimpan kode dan juga 2 KB yang digunakan untuk *bootloader* [15].

2.2.3.2 Sensor MQ-7

Sensor MQ-7 adalah sensor gas yang digunakan dalam peralatan untuk mendeteksi gas CO dalam kehidupan sehari-hari, industri, atau mobil. Sensor MQ-7 memiliki keunggulan yaitu memiliki sensitivitas yang tinggi terhadap gas CO. Sensor ini juga memiliki stabilitas yang baik dan umur pemakaian yang panjang. Sensor MQ-7 terdiri dari bahan *mikrotube* keramik AL_2O_3, lapisan sensitif SnO_2, elektroda, dan pemanas dalam cangkang plastik dan baja tahan karat. MQ-7 memiliki 4 pin: 2 untuk sinyal dan 2 untuk suplai arus [16].

Gambar 2.2 Sensor MQ-7 [16]

Gambar 2.2 merupakan modul sensor MQ-7. Pin pada sensor MQ-7 meliputi pin VCC yang dihubungkan dengan tegangan suplai 5V dari Arduino. Kemudian pin *ground* (GND) dihubungkan dengan pin GND dari Arduino. Selanjutnya pada pin A0 yaitu *output* analog yang terhubung ke pin analog pada

Arduino. Cara kerja dari sensor MQ-7 yaitu elemen pemanas internal dan elemen penginderaan akan berubah resistansinya ketika terpapar gas CO. Elemen pemanas memastikan bahwa sensor berada pada suhu operasi yang tepat, sedangkan elemen penginderaan mendeteksi perubahan konsentrasi gas. Perubahan resistansi ini menghasilkan tegangan *output* analog yang dapat dibaca oleh mikrokontroler seperti Arduino untuk menentukan konsentrasi gas di udara [16].

Tabel 2.3 Spesifikasi Sensor MQ-7 [16]

Rentang Deteksi	20-2000 ppm
Tegangan Operasi	5V DC
Arus Operasi	< 150 mA
Waktu Pemanasan	2.5 menit
Output	Analog
Waktu Respon	< 10 detik
Daya Pemanasan	0.5 W

Tabel 2.3 merupakan spesifikasi dari sensor MQ-7. Sensor MQ-7 memiliki rentang deteksi gas CO dari 20 ppm hingga 2000 ppm. Tegangan operasi pada sensor MQ-7 sebesar 5V DC. Kemudian arus operasi pada snsor MQ-7 sebesar kurang dari 150 mA. Waktu Pemanasan atau *pre-heating* yang dibutuhkan sensor MQ-7 sekitar 2.5 menit. Waktu yang dibutuhkan sensor MQ-7 untuk merespon gas CO selama kurang dari 10 detik. Kemudian daya pemanasan pada sensor MQ-7 sebesar 0.5 W. *Output* yang dihasilkan pada sensor MQ-7 yaitu *output* analog yang bervariasi sesuai dengan gas yang terdeteksi [16].

2.2.3.3 *Liquid Crystal Display* (LCD) 16x2

LCD 16x2 adalah modul layar kristal cair yang sering digunakan dalam berbagai proyek elektronik untuk menampilkan informasi. LCD 16x2 dihubungkan dengan adapter *Inter-Integrated Circuit* (I2C) untuk komunikasi serial dua arah, yang berfungsi sebagai pengiriman dan penerimaan data. LCD tersedia dalam bentuk modul yang mempunyai pin data, control catu daya, dan pengatur kontras. Keunggulan modul LCD serial I2C ini adalah mengurangi koneksi sirkuit, menghemat banyak pin I/O pada papan Arduino, dan sangat menyederhanakan

pengembangan *firmware* menggunakan perpustakaan Arduino yang tersedia. Pada baris *Liquid Crystal* (2, 3, 4, 5, 6, 12, 7) terdapat proses inisialisasi yang terhubung ke pin LCD RS, *Enable*, D4, D5, D6, dan D7 [17].

Gambar 2.3 *Liquid Crystal Display* 16x2 dan I2C [17]

Gambar 2.3 merupakan modul LCD 16x2 dengan serial I2C. LCD berfungsi untuk menampilkan karakter, huruf, atau gambar dengan menggunakan kristal cair. Arti "16x2" menunjukkan bahwa layar ini memiliki 16 kolom dan 2 baris, sehingga dapat menampilkan hingga 32 karakter (16 karakter per baris). LCD 16x2 adalah pilihan populer karena harganya yang terjangkau, konsumsi daya yang rendah. Modul LCD 16x2 sangat berguna untuk menampilkan informasi dalam proyek Arduino [18].

Tabel 2.4 Spesifikasi LCD 16x2 [17]

Operasi Tegangan	5V
Jumlah Kolom dan Baris	16 Kolom x 2 Baris
Kontrol Kontras	Potensiometer
Kontrol Lampu	*Firmware*
Ukuran *Board*	80x36 mm

Tabel 2.4 merupakan spesifikasi dari LCD 16x2. LCD 16x2 membutuhkan operasi tegangan sebesar 5V DC. Jumlah kolom dan baris pada LCD 16x2 berjumlah 16 kolom dan 2 baris yang mana dapat menampilkan hingga 32 karakter. LCD 16x2 dapat diatur kontrasnya dengan menggunakan potensiometer bawaan pada modul LCD. Kontrol lampu pada LCD 16x2 menggunakan *firmware* atau

dalam bentuk kode program yang dituliskan pada *software*. LCD 16x2 memiliki ukuran board sebesar 80x36 mm [17].

2.2.3.4 *Buzzer*

Buzzer merupakan sebuah komponen elektronika yang berfungsi untuk mengubah getaran listrik menjadi getaran suara. *Buzzer* memiliki 2 pin yang sangat kecil dan efisien sehingga mudah digunakan pada *breadboard* dan pada PCB. Maka dari itu *buzzer* menjadi komponen yang banyak digunakan di sebagian besar aplikasi elektronik. Buzzer dapat dinyalakan dengan menggunakan power DC mulai dari 4V hingga 9V. Biasanya *buzzer* dikaitkan dengan rangkaian *switching* untuk menghidupkan atau mematikan *buzzer* pada waktu dan interval yang diperlukan [19].

Gambar 2.4 *Buzzer* [20]

Gambar 2.4 merupakan modul *buzzer* yang dapat mengeluarkan getaran suara. Pada dasarnya prinsip kerja *buzzer* hampir sama dengan loud speaker. *Buzzer* terdiri dari kumparan yang terpasang pada diafragma dan kemudian kumparan tadi akan tertarik ke dalam atau keluar tergantung dari arah arus polaritas magnetnya, maka setiap gerakan kumparan akan menggerakan diafragma secara bolak-balik sehingga membuat udara bergetar yang akan menghasilkan suara. *Buzzer* biasa digunakan sebagai indikator bahwa proses telah selesai atau terjadi suatu kesalahan pada sebuah alat (alarm) [20].

2.2.3.5 *Light Emitting Diode* (LED)

LED adalah sebuah komponen yang berfungsi sebagai pemancar cahaya. Struktur dari LED mirip dengan struktur dioda yaitu semikonduktor. LED dikatakan memiliki efisiensi cahaya yang tinggi. Chip dari LED umumnya sama dengan chip dari dioda. Pemberian tegangan yang terlalu besar terhadap LED dapat merusak atau bisa membakar LED. LED beroperasi mirip dengan dioda dengan dua kutub, yaitu positif (P) dan negatif (N) [21].

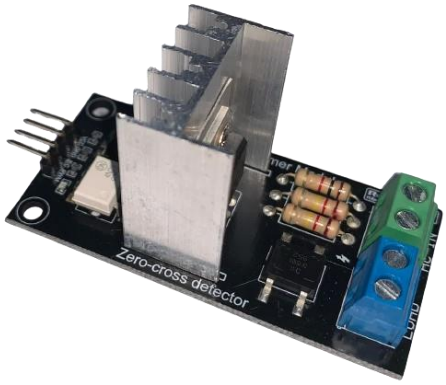
Gambar 2.5 LED [21]

Gambar 2.5 merupakan LED untuk pemancar cahaya. Prinsip Kerja dari LED yaitu bekerja berdasarkan prinsip elektroluminesensi, dimana bahan semikonduktor memancarkan cahaya ketika elektron dan hole (lubang elektron) bergabung kembali. Ketika arus listrik melewati LED, elektron di semikonduktor berpindah dari pita konduksi ke pita valensi dan memancarkan energi dalam bentuk cahaya. LED memiliki kelebihan yang meliputi efisiensi energi, memiliki ukuran kecil sehingga mudah untuk diintegrasikan ke dalam berbagai perangkat dan memiliki respon yang cepat. Penggunaan LED dapat diterapkan sebagai indikator pada alat elektronik, sebagai *display* dan dapat digunakan pada aplikasi medis seperti terapi cahaya [21].

2.2.3.6 *Dimmer* AC

Dimmer AC adalah modul untuk mengontrol tegangan arus bolak-balik yang dengan tegangan *input* 220V/110V. Pada Arduino, *dimmer* dikontrol menggunakan perpustakaan RBDdimmer.h. Dengan menggunakan interupsi *runtime* dan eksternal, maka membuat kode lebih mudah untuk ditulis dan

meningkatkan waktu pemrosesan kode utama. Beberapa *dimmer* dapat dikontrol melalui mikrokontroler. *Dimmer* dihubungkan ke pengontrol Arduino dengan menggunakan dua pin digital [22].



Gambar 2.6 *Dimmer* AC [22]

Gambar 2.6 merupakan modul *dimmer* AC. Dimmer AC dapat digunakan untuk mengontrol tingkat kecerahan lampu atau kecepatan motor dengan cara mengatur daya yang disuplai ke perangkat tersebut. Dimmer AC biasanya digunakan dalam aplikasi pencahayaan rumah tangga untuk mengatur intensitas cahaya lampu pijar atau lampu halogen, serta untuk mengontrol kecepatan motor AC, seperti kipas angin dan *exhaut fan*. Komponen utama dimmer AC meliputi TRIAC yaitu komponen semikonduktor yang mengontrol aliran daya ke beban. Diac yaitu Komponen yang membantu menyalakan TRIAC. Potensiometer berfungsi untuk mengatur waktu nyala TRIAC dan mengatur kecerahan atau kecepatan motor. Kapasitor dan Resistor yaitu komponen yang membentuk rangkaian timing untuk menyalakan TRIAC pada titik tertentu dalam gelombang AC [22].

2.2.3.7 *Exhaust Fan* AC

Exhaust fan merupakan kipas yang memiliki fungsi untuk menjaga kebersihan udara didalam ruangan. Mekanisme *exhaust fan* adalah mengambil udara dari dalam ruangan dan mengirimkannya ke luar, sekaligus membawa udara

segar dari luar ke dalam ruangan. Untuk menjaga kualitas udara bersih, diperlukan sirkulasi udara yang senantiasa mengganti udara didalam ruangan dengan udara luar yang segar [23].

Gambar 2.7 *Exhaust fan* [23]

Gambar 2.7 merupakan *Exhaust Fan* AC yang dapat digunakan di dapur, kamar mandi, dan pada kendaraan mobil yang berfungsi untuk mengontrol kualitas udara dan mencegah penumpukan panas atau kelembapan yang berlebihan. Fungsi dan manfaat *Exhaust Fan* dapat digunakan sebagai ventilasi udara untuk mengeluarkan udara yang lembap, berbau, atau tercemar dari dalam ruangan. Sebagai pengendalian kelembapan untuk mengurangi kelembapan berlebih yang dapat menyebabkan jamur dan lumut. Kemudian untuk pengendalian suhu dengan membantu menurunkan suhu dalam ruangan dengan mengeluarkan udara panas. Kemudian untuk meningkatkan kualitas udara agar memastikan aliran udara segar dari luar dan meningkatkan kualitas udara dalam ruangan. Mengurangi polusi udara dalam ruangan untuk mengeluarkan asap, bau, dan partikel polutan lainnya [23].

2.2.4 *Software*

2.2.4.1 Arduino *Integrated Development Environment* (IDE)

Arduino IDE adalah *software* yang berfungsi untuk memprogram dan mengunggah kode ke papan Arduino. *Software* ini sering digunakan oleh para

pecinta elektronika untuk mengembangkan berbagai macam proyek atau penelitian yang berbasis Arduino. Fitur utama dari *software* Arduino IDE meliputi pemrograman bahasa C/C++, pustaka, editor kode, *compiler* dan *uploader*, monitor serial, dukungan untuk berbagai board, dan pemecahan masalah [24].

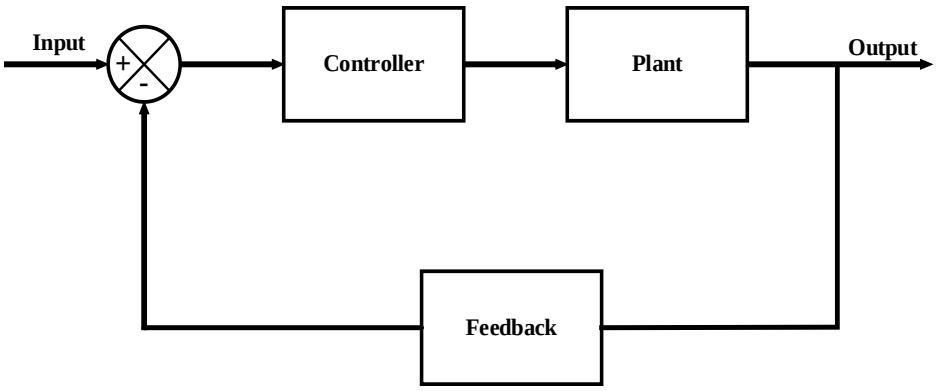
Gambar 2.8 Arduino IDE [25]

Gambar 2.8 merupakan tampilan awal *software* Arduino IDE yang memiliki fungsi untuk menulis, mengedit, mengompilasi, dan mengunggah kode ke papan mikrokontroler Arduino. Arduino IDE dirancang untuk memudahkan pengguna, terutama pemula, dalam menulis dan mengunggah kode ke papan Arduino tanpa memerlukan pengetahuan mendalam tentang bahasa pemrograman atau elektronika yang kompleks. Program Arduino dibuat menggunakan Arduino IDE dan disebut sebagai sketsa. Sketsa ditulis dalam editor teks dan disimpan dengan ekstensi file .ino. [25].

2.2.5 Proporsional Integral Derivatif (PID)

PID adalah metode kontrol yang sangat umum digunakan dalam sistem kontrol otomatis untuk mencapai tujuan tertentu, seperti mempertahankan suhu, posisi motor servo, kecepatan kipas, atau variabel lain pada nilai yang diinginkan. Fungsi pengontrol PID adalah menghitung dan mengendalikan daya kendali berdasarkan selisih antara setpoint dan nilai aktual yang diukur oleh sistem. Selain itu, pengontrol ini dapat dengan mudah dikombinasikan dengan metode pengendalian lain seperti *fuzzy* dan *robust*. Pengontrol PID sebenarnya merupakan kombinasi dari tiga tipe parameter yaitu pengontrol P (proporsional), pengontrol I

(integral), dan pengontrol D (derivatif). Setiap fungsi memerlukan parameter tertentu yang disebut konstanta [26].



Gambar 2.9 Blok Diagram *Unity Feedback System* [26]

Gambar 2.9 merupakan blok diagram *unity feedback system*. Pada bagian *input* yaitu nilai *set point* yang diinginkan. Kemudian terdapat *error detector* yang berfungsi untuk membandingkan nilai set point dengan nilai umpan balik. Hasil dari *error detector* akan menghasilkan kesalahan. Selanjutnya terdapat *controller* yaitu dengan menggunakan PID yang mana *controller* tersebut akan menerima nilai kesalahan dan akan dihitung berdasarkan tiga parameter yaitu proporsional (P), integral (I) dan derivatif (D) yang akan menghasilkan *output* kontrol. Kemudian hasil *output* kontrol akan diterima oleh *plant* yang selanjutnya diterapkan kontrol ke sistem *plant* untuk menghasilkan perubahan yang diinginkan dalam variabel kontrol. Kemudian terdapat *feedback* yang berfungsi untuk menerima hasil *output* aktual dari *plant* dan kemudian nilai tersebut digunakan sebagai masukan ke sensor untuk dibandingkan kembali dengan *setpoint* [26].

Tabel 2.5 Respon PID Terhadap Perubahan Konstanta [26]

Closed-Loop Response	***Rise Time***	***Overshoot***	***Settling Time***	***SS Error***
K_p	Menurunkan	Menaikkan	Perubahan Kecil	Menurunkan
K_i	Menurunkan	Menaikkan	Menurunkan	Mengeliminasi
K_d	Perubahan Kecil	Menurunkan	Menurunkan	Perubahan Kecil

Tabel 2.5 merupakan respon PID terhadap perubahan konstanta yang digunakan sebagai panduan hanya jika terjadi perubahan yang sedang berlangsung. Pada konstanta proporsional (K_p) berfungsi untuk mengatur kekuatan respons terhadap kesalahan yang dihasilkan. Apabila nilai K_p kecil maka respon sistem akan lebih lambat, kemudian jika nilai Kp besar maka respon sistem akan lebih cepat dan lebih agresif. Pada konstanta integral (K_i) berfungsi untuk mengatur respons berdasarkan akumulasi kesalahan dari waktu ke waktu dan akan membantu menghilangkan *error steady state*. Jika nilai K_i kecil maka efek integral akan lebih lemah sehingga sistem akan menghilangkan kesalahan *steady-state* lebih lambat, kemudian apabila nilai K_i besar maka efek integral akan lebih kuat sehingga membantu menghilangkan kesalahan *steady-state* lebih cepat. Kemudian pada konstanta derivatif (K_d) berfungsi untuk mengatur respons berdasarkan kecepatan perubahan kesalahan, mengurangi overshoot dan meningkatkan stabilitas sistem. Apabila nilai K_d kecil maka efek derivatif akan lebih lemah sehingga kemampuan untuk meredam osilasi dan *overshoot* akan berkurang, kemudian jika nilai K_d besar maka efek derivatif akan lebih kuat sehingga membantu meredam osilasi dan *overshoot*. Nilai K_p, K_i, dan K_d mengacu pada domain waktu respon sistem dengan perhitungan besar nilai error. Berdasarkan hal tersebut untuk menentukan nilai *u(t)* digunakan persamaan sebagai berikut: [26].

$$u(t) = K_p e(t) + K_i \int e(t)\ dt + K_d \frac{de(t)}{dt} \qquad \textbf{(2.1)}$$

Di mana:

u(t) = *Output* kontrol pada waktu t

K_p = Konstanta Proporsional

K_i = Konstanta Integral

K_d = Konstanta Derivatif

e(t) = Kesalahan pada waktu t

$\int e(t)\ dt$ = Integral kesalahan terhadap waktu

$\frac{de(t)}{dt}$ = Derivatif dari kesalahan terhadap waktu [26].

Untuk menghitung parameter PID, ada beberapa istilah yang digunakan yaitu :

- *Set point* adalah nilai yang diinginkan atau target dari suatu variabel proses dalam sistem kontrol.
- *Steady state* adalah keadaan dalam suatu sistem dinamis di mana variabel-variabel dalam sistem tidak berubah lagi seiring waktu atau berubah dengan laju konstan.
- *Error steady state* adalah kondisi pada saat sistem kondisi respon sinyal sudah atau belum mencapai *steady state* tetapi terdapat *error*.
- *Overshoot* adalah kondisi di mana respon sistem melebihi nilai setpoint sebelum akhirnya menetap pada nilai *setpoint* atau stabil.
- *Time Rise* adalah waktu yang diperlukan oleh *output* sistem untuk pertama kali mencapai *steady state*.
- *Time Peak* adalah waktu yang dibutuhkan oleh respon sistem untuk mencapai nilai puncak pertama setelah mencapai *setpoint*.
- *Time Settling* adalah waktu yang diperlukan oleh respon sistem untuk mencapai nilai stabil [27].

2.2.5.1 Kontrol Proporsional (P)

Kontrol proporsional adalah jenis kontroler yang menggunakan kesalahan antara setpoint dan nilai proses aktual untuk menentukan output kontrol. Kontrol Proporsional (P) merupakan pengembangan lebih lanjut dari kontrol dua titik (*on*/*off*). Kontroler ini bertujuan untuk mengurangi kesalahan dengan menyesuaikan *output* secara proporsional terhadap besarnya kesalahan. Salah satu bentuk paling sederhana dari kontroler *feedback* dan sering digunakan dalam berbagai aplikasi kontrol industri. Untuk mengetahui *output* kontrol atau *u(t)* dari kontrol proporsional, digunakan persamaan berikut: [28].

$$u(t) = K_p\, e(t) \tag{2.2}$$

Di mana:

u(t) = *Output* kontrol pada waktu t

K_p = Konstanta Proporsional

e(t) = Kesalahan pada waktu t [28].

2.2.5.2 Kontrol Integral

Kontrol integral adalah salah satu komponen dari kontrol PID yang bertujuan untuk menghilangkan kesalahan steady-state dalam sistem kontrol. Fungsi pengontrol integral adalah menghasilkan sebuah respon sistem yang bebas dari kesalahan kondisi tunak atau *steady state*. Dengan menggunakan pengontrol integral dapat meningkatkan respons sistem dan mengurangi kesalahan kondisi tunak hingga nol. Kontrol integral menambahkan semua kesalahan dari waktu ke waktu, yang berarti jika ada kesalahan kecil yang tetap ada, kontroler akan terus menambah output kontrol untuk mengoreksi kesalahan tersebut sampai kesalahan menjadi nol. Ini efektif untuk menghilangkan kesalahan steady-state, yang tidak dapat dihilangkan oleh kontrol proporsional saja. Untuk mengetahui output kontrol atau *u(t)* dari kontrol integral, digunakan persamaan berikut: [28].

$$u(t) = \int_0^t e(t)\, dt \qquad \textbf{(2.3)}$$

Di mana:

u(t) = *Output* kontrol pada waktu t

K_i = Konstanta Integral

$\int e(t)\ dt$ = Integral kesalahan terhadap waktu [28].

2.2.5.3 Kontrol Derivatif

Kontrol derivatif adalah komponen dalam kontrol PID yang berfungsi untuk memperkirakan arah dan kecepatan perubahan kesalahan. Kontrol derivatif bertujuan untuk mengurangi osilasi dan overshoot dengan memberikan respons berdasarkan laju perubahan kesalahan. Kontrol derivatif memberikan koreksi berdasarkan kecepatan perubahan kesalahan. Jika kesalahan meningkat dengan cepat, kontrol derivatif akan menghasilkan output yang besar untuk mencegah sistem dari overshoot atau berosilasi terlalu banyak sehingga membantu sistem untuk mencapai stabilitas lebih cepat. Untuk mengetahui output kontrol atau *u(t)* dari kontrol derivatif, digunakan persamaan berikut: [28].

$$u(t) = K_d \frac{de(t)}{dt} \qquad \textbf{(2.4)}$$

Di mana:

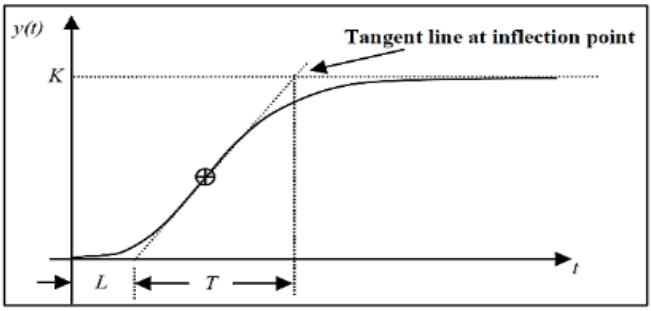
$u(t)$ = *Output* kontrol pada waktu t

K_d = Konstanta Derivatif

$\frac{de(t)}{dt}$ = Derivatif dari kesalahan terhadap waktu [28].

2.2.6 Metode *Ziegler-Nichols*

Metode *Ziegler-Nichols* adalah prosedur penyetelan parameter kontrol PID yang diperkenalkan oleh John G. Ziegler dan Nathaniel B. Nichols pada tahun 1942. *Ziegler-Nichols* merupakan metode *tunning* parameter kontrol PID dengan menguji secara eksperimental respon suatu sistem terhadap input dalam satuan langkah dan membuat kurva S. Metode ini digunakan untuk menemukan parameter K_p (konstanta proporsional), K_i (konstanta integral), dan K_d (konstanta derivatif) yang optimal untuk mencapai kinerja yang diinginkan dalam sistem kontrol. Ada dua pendekatan utama dalam metode Ziegler-Nichols yaitu metode *step response method* dan *tunning ultimate gain method* [29].



Gambar 2.10 Kurva Reaksi *Ziegler-Nichols* [29]

Gambar 2.10 merupakan kurva reaksi *ziegler nichols* atau disebut dengan kurva S. Terdapat dua parameter yaitu waktu tunda (L) yaitu waktu yang dibutuhkan respon sistem untuk pertama kali mulai naik setelah input langkah diberikan. Kemudian parameter konstanta waktu (T) yaitu waktu yang dibutuhkan sistem untuk mencapai 63% dari respon akhirnya. Parameter L dan T dapat diperoleh dengan cara menggambar garis singgung pada titik belok kurva S dan mencari perpotongannya [29]

Tabel 2.6 Metode *Tunning Ziegler-Nichols* [29]

Type of controller	Kp	Ti	Td
P	T / L	∞	0
PI	0.9 T / L	L / 0.3	0
PID	1.2 T / L	2L	0.5L

Tabel 2.6 merupakan metode tunning *ziegler nichols* untuk mencari parameter PID. Untuk mencari parameter PID, sebelumnya harus menentukan parameter dari respon sistem yaitu T dan L. Parameter PID yang harus dicari meliputi nilai K_p, T_i dan T_d. Untuk mencari nilai K_p yaitu dengan menggunakan rumus 1,2 *T/L*. Kemudian untuk mencari nilai T_i yaitu dengan menggunakan rumus 2*L*. Selanjutnya untuk mencari nilai T_d menggunakan rumus 0.5*L*. Setelah mendapatkan nilai K_p, T_i, dan T_d maka dapat mencari nilai K_i dan K_d. Rumus persamaan untuk mencari nilai K_i yaitu K_p / T_i. Kemudian rumus persamaan untuk mencari nilai K_d yaitu $K_p T_d$ [29].