Copilot

Vamos considerar um caso prático envolvendo cálculos em sistemas embarcados: o controle de um motor usando um controlador PID (Proporcional-Integral-Derivativo).

Caso de Estudo: Controle de Motor com PID

Contexto: Um sistema embarcado é usado para controlar a velocidade de um motor DC. O objetivo é manter a velocidade do motor constante, mesmo quando há variações na carga.

Componentes:

- **Microcontrolador**: Arduino Uno
- **Sensor de Velocidade**: Encoder
- **Atuador**: Motor DC
- **Controlador**: PID

Cálculos Envolvidos:

1. **Erro de Velocidade**:
    - O erro é a diferença entre a velocidade desejada (V_{d}) e a velocidade atual (V_{a}).
    - $$ \text{Erro} = V_{d} - V_{a} $$
2. **Termo Proporcional (P)**:
    - Proporcional ao erro atual.
    - $$ P = K_{p} \times \text{Erro} $$
    - Onde K_{p} é o ganho proporcional.
3. **Termo Integral (I)**:
    - Proporcional à soma dos erros passados (acumulação do erro).
    - $$ I = K_{i} \times \sum \text{Erro} \times \Delta t $$
    - Onde K_{i} é o ganho integral e Δt é o intervalo de tempo entre as medições.
4. **Termo Derivativo (D)**:
    - Proporcional à taxa de variação do erro.
    - $$ D = K_{d} \times \frac{\text{d(Erro)}}{\text{d}t} $$
    - Onde K_{d} é o ganho derivativo.
5. **Saída do Controlador PID**:
    - A saída é a soma dos três termos.

- $$ \text{Saída} = P + I + D $$

Implementação no Arduino:

```
// Definição dos ganhos do PID
double Kp = 2.0;
double Ki = 0.5;
double Kd = 1.0;

// Variáveis para armazenar os valores do PID
double erro, erroAnterior, integral, derivativo, saida;

// Variáveis de tempo
unsigned long tempoAnterior, tempoAtual;
double deltaT;

// Velocidades
double velocidadeDesejada = 100.0; // Exemplo: 100 RPM
double velocidadeAtual;

void setup() {
  // Configurações iniciais
  tempoAnterior = millis();
}

void loop() {
  // Atualiza o tempo
  tempoAtual = millis();
  deltaT = (tempoAtual - tempoAnterior) / 1000.0; // Converte para segundos

  // Leitura da velocidade atual do motor
  velocidadeAtual = lerVelocidadeMotor();

  // Calcula o erro
  erro = velocidadeDesejada - velocidadeAtual;

  // Calcula os termos do PID
  integral += erro * deltaT;
  derivativo = (erro - erroAnterior) / deltaT;

  // Calcula a saída do PID
  saida = (Kp * erro) + (Ki * integral) + (Kd * derivativo);

  // Aplica a saída ao motor
  controlarMotor(saida);

  // Atualiza o erro anterior e o tempo
  erroAnterior = erro;
  tempoAnterior = tempoAtual;
}

double lerVelocidadeMotor() {
  // Função fictícia para ler a velocidade do motor
  return 90.0; // Exemplo: 90 RPM
}

void controlarMotor(double valor) {
  // Função fictícia para controlar o motor
  // Aqui você aplicaria o valor calculado ao motor
}
```

Explicação dos Cálculos

- **Erro**: A diferença entre a velocidade desejada e a atual.
- **Termo Proporcional**: Ajusta a saída com base no erro atual.
- **Termo Integral**: Ajusta a saída com base na soma dos erros passados, ajudando a eliminar o erro residual.
- **Termo Derivativo**: Ajusta a saída com base na taxa de variação do erro, ajudando a prever e corrigir rapidamente mudanças no erro.

Este exemplo ilustra como os cálculos matemáticos são aplicados em sistemas embarcados para controlar um motor de forma precisa e eficiente. Se precisar de mais detalhes ou tiver outras perguntas, estou aqui para ajudar!