

## **Tribômetro**

Gustavo Marinho Barbosa dos Santos (537453) E-mail: [gustavomarinho@alu.ufc.br](mailto:gustavomarinho@alu.ufc.br)  
Helio Gouveia Medeiros Neto (Matrícula 553274) E-mail: [heliogmedeiros@alu.ufc.br](mailto:heliogmedeiros@alu.ufc.br)  
Igor Damião de Oliveira Pereira (Matrícula 553883) E-mail: [igordamiao@alu.ufc.br](mailto:igordamiao@alu.ufc.br)  
Joel da Silva Teixeira (Matrícula 547778) E-mail: [joeldsteixeira@alu.ufc.br](mailto:joeldsteixeira@alu.ufc.br)  
José Milton Soares de Castro Filho (Matrícula 538374) E-mail: [miltonfilho@alu.ufc.br](mailto:miltonfilho@alu.ufc.br)  
José Ronald Gomes Pereira (542230) E-mail: [ronaldgomes@alu.ufc.br](mailto:ronaldgomes@alu.ufc.br)

### **1. Introdução**

A Tribologia, termo cunhado em 1966 pelo comitê da OCDE (Organização para a Cooperação e Desenvolvimento Econômico), é a ciência e tecnologia que investiga a interação de superfícies em movimento relativo, abrangendo os fenômenos de atrito, desgaste e lubrificação. A relevância desse campo é imensa: estima-se que mais de 20% da energia consumida mundialmente seja utilizada para vencer o atrito em sistemas mecânicos, e uma parcela significativa de falhas em máquinas ocorre devido ao desgaste prematuro de componentes (BHUSHAN, 2013). Portanto, o domínio desses conceitos é crucial para a engenharia moderna, visando o aumento da eficiência energética e a sustentabilidade dos processos industriais.

No contexto acadêmico, o ensino das leis de atrito e da mecânica de contato é frequentemente abordado de maneira teórica e idealizada. Embora os livros didáticos apresentem modelos matemáticos fundamentais, a visualização prática da transição entre o atrito estático e cinético, ou o efeito "stick-slip" (agarra-eskorrega), exige experimentação. Entretanto, as instituições de ensino enfrentam um obstáculo financeiro: os tribômetros comerciais — equipamentos destinados a medir forças de atrito e desgaste — possuem custos elevados, muitas vezes na casa dos milhares de dólares, e interfaces complexas voltadas para pesquisa avançada, o que inviabiliza seu uso massivo em laboratórios de graduação (LUDEMA, 1996).

Diante dessa lacuna entre a teoria ensinada e a prática acessível, surge a oportunidade de aplicar conceitos de Engenharia Reversa e o movimento "Maker" para o desenvolvimento de ferramentas didáticas. O advento de tecnologias de prototipagem rápida e plataformas de eletrônica embarcada de baixo custo, como o Arduino, permitiu que instrumentos de medição precisos fossem construídos com orçamentos reduzidos.

Este projeto tem como objetivo principal o desenvolvimento, construção e validação de um tribômetro automatizado de baixo custo. O equipamento visa realizar a aquisição de dados de força de atrito em tempo real, permitindo a análise gráfica do comportamento de diferentes pares de materiais. A iniciativa justifica-se pela necessidade de democratizar o acesso a ensaios experimentais de qualidade, fornecendo aos estudantes uma ferramenta que integra conceitos de mecânica clássica, instrumentação eletrônica e programação.

### **2. Fundamentação Teórica**

A correta interpretação dos dados gerados pelo tribômetro exige uma base sólida sobre a física do contato e os princípios de medição eletrônica utilizados no projeto.

## 2.1 A Natureza da Força de Atrito

O atrito é definido como a força resistiva tangencial que atua na interface comum de dois corpos em contato, opondo-se ao movimento relativo ou à tendência de movimento. Embora as Leis de Amontons-Coulomb (séculos XVII e XVIII) descrevam o atrito macroscópico de forma satisfatória para muitas aplicações de engenharia, a origem do fenômeno é microscópica.

Segundo a teoria da adesão, proposta por Bowden e Tabor (1950), as superfícies reais nunca são perfeitamente lisas; elas possuem irregularidades chamadas asperezas. O contato real ocorre apenas nos picos dessas asperezas, fazendo com que a área real de contato seja uma fração minúscula da área aparente. O atrito surge, fundamentalmente, de dois mecanismos:

1. **Adesão:** A formação de ligações moleculares (microsoldas) nos pontos de contato, que precisam ser cisalhadas para que ocorra movimento.
2. **Deformação:** O sulcamento de um material mais duro sobre um mais macio (componente abrasivo).

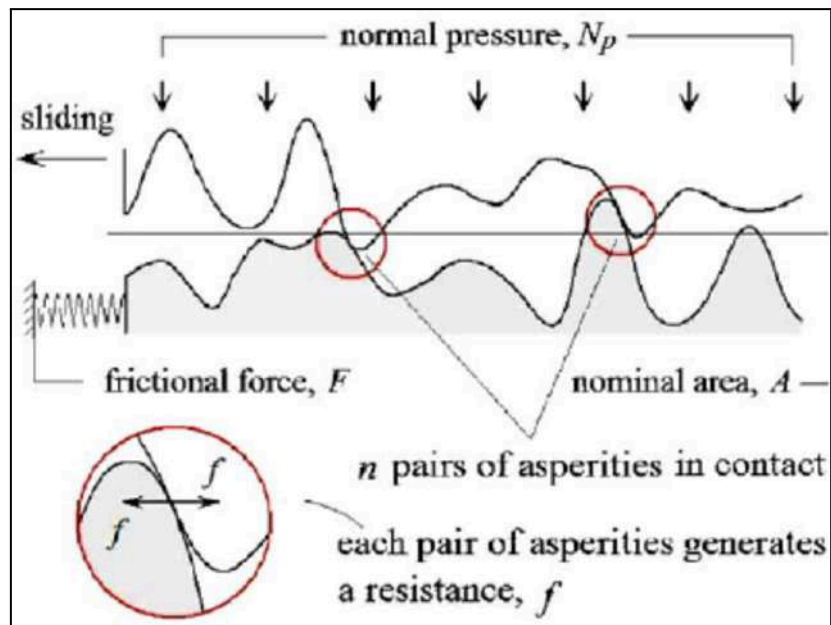


Figura 1 - Representação esquemática do contato entre asperezas de duas superfícies

## 2.2 Regimes de Atrito: Estático e Cinético

O comportamento da força de atrito ( $F_{at}$ ) varia conforme o estado de movimento do corpo.

O Atrito Estático ocorre quando as superfícies estão em repouso relativo. É uma força de reação que cresce linearmente conforme a força externa aplicada aumenta, até atingir um ponto crítico denominado "iminência do movimento". Neste ponto, a força de atrito estático é máxima e dada por:

$$F_{s, \text{máx}} = \mu_s \cdot N$$

Onde  $\mu_s$  é o coeficiente de atrito estático e  $N$  é a força normal. Se a força aplicada exceder  $Fs, máx$ , as microssoldas entre as asperezas se rompem e o movimento se inicia.

O **Atrito Cinético** (ou Dinâmico) atua durante o deslizamento relativo. Devido à inércia e ao fato de que as asperezas não têm tempo suficiente para formar ligações tão fortes quanto no repouso, a força necessária para manter o movimento é menor que a necessária para iniciá-lo. Assim, experimentalmente, verifica-se que  $\mu_k < \mu_s$  (Halliday; Resnick, 2012).

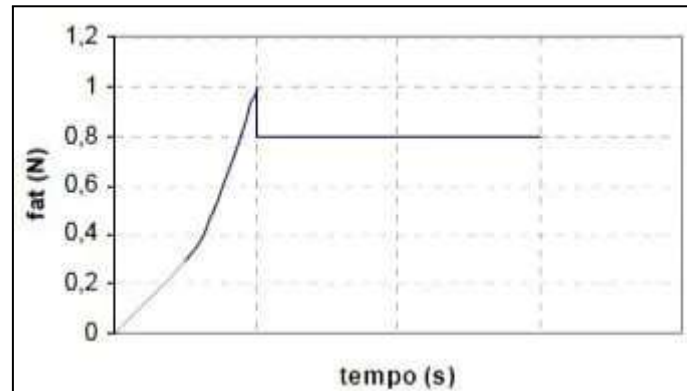


Figura 2 - Representação esquemática do contato entre asperezas de duas superfícies

### 2.3 Trabalho da Força de Atrito e Dissipação de Energia

Na mecânica, o trabalho ( $W$ ) é a medida da energia transferida pela aplicação de uma força ao longo de um deslocamento. Como a força de atrito cinético atua sempre no sentido oposto ao vetor velocidade (e deslocamento), o trabalho realizado por ela é negativo, caracterizando um sistema dissipativo.

$$W_{fat} = \int_{xi}^{xf} - Fk dx = - \mu_k \cdot N \cdot \Delta x$$

Essa "perda" de energia mecânica não viola a Lei da Conservação de Energia (Primeira Lei da Termodinâmica). O trabalho realizado pelo atrito é convertido irreversivelmente em energia interna ( $U$ ) das superfícies em contato, manifestando-se como aumento de temperatura (calor) e deformação plástica microscópica (desgaste). Em projetos de engenharia, quantificar esse trabalho é essencial para dimensionar sistemas de refrigeração e prever a vida útil de componentes sujeitos à fadiga por contato (NORTON, 2013).

### 2.4 Aquisição de Sinais e Instrumentação

Para validar experimentalmente as teorias acima, o tribômetro precisa converter grandezas físicas (força) em dados digitais. Este processo é denominado Aquisição de Sinais.

O sistema proposto utiliza uma **Célula de Carga** como transdutor primário. Células de carga operam baseadas em extensômetros (*strain gauges*), que variam sua resistência elétrica conforme sofrem deformação mecânica. Como essa variação de

resistência é extremamente sutil, o sinal elétrico de saída é muito baixo (na ordem de milivolts).

Para ler esse sinal, é necessário um circuito de condicionamento. O módulo **HX711** é frequentemente utilizado nesse contexto, pois integra um amplificador de baixo ruído e um conversor Analógico-Digital (A/D) de 24 bits. Isso garante a precisão necessária para detectar as pequenas variações de força durante o ensaio de atrito (BOLTON, 2010). A taxa de amostragem (frequência com que o sensor lê os dados) é um fator crítico: ela deve ser alta o suficiente para capturar o pico do atrito estático antes que ele caia para o regime cinético.

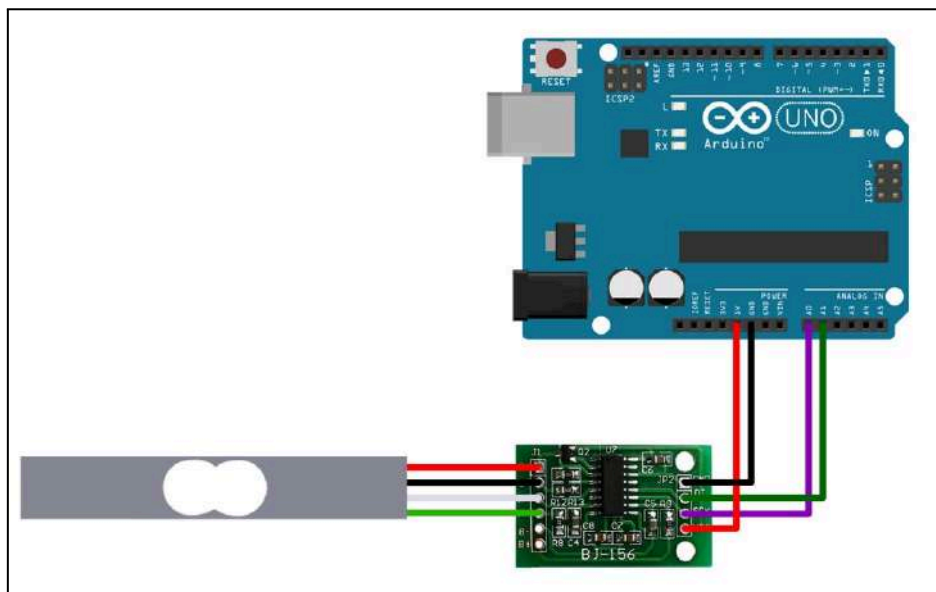


Figura 3 - Sistema de aquisição de sinais

## 2.5 Microcontrolador Arduino

A central de controle do tribômetro é a plataforma Arduino. Trata-se de uma plataforma de prototipagem eletrônica *open-source* baseada em hardware e software flexíveis. O modelo utilizado, tipicamente o Arduino Uno, baseia-se no microcontrolador ATmega328P.

O Arduino desempenha três funções vitais no projeto:

1. **Controle:** Pode acionar motores (se o tribômetro for motorizado) para garantir uma velocidade de deslizamento constante.
2. **Processamento:** Recebe os dados digitais do módulo HX711, aplica fatores de calibração conhecidos e converte os bits em unidades de força (Newtons).
3. **Comunicação:** Envia os dados processados via porta Serial (USB) para um computador, onde softwares como o *Serial Plotter* ou Excel podem gerar os gráficos em tempo real.

A escolha do Arduino deve-se à sua robustez, baixo custo e à vasta comunidade de suporte, o que facilita a implementação de códigos (firmware) para leitura de sensores complexos (MCROBERTS, 2011).

## 2.6 Projeto CAD

O desenvolvimento do CAD do equipamento se deu em 3 etapas: Estudo de caso, definição dos materiais e, por fim, desenvolvimento do projeto. Durante a primeira fase foi realizado um estudo teórico referente aos principais componentes presentes no equipamento identificando os possíveis materiais necessários para a fabricação. Diante dos materiais estipulados, o grupo optou por utilizar placas de fibras de madeiras (MDF) como único material, para a base e a rampa do tribômetro, devido principalmente à facilidade de fabricação e ao baixo custo de aquisição.

Os componentes foram projetados considerando principalmente dois fatores: facilidade de estudo das amostras, permitindo uma movimentação ininterrupta por parte do corpo de prova de 50mm x 50mm, e inclinação mínima da rampa de 45°. Todas as dimensões estão representadas nas figuras X.

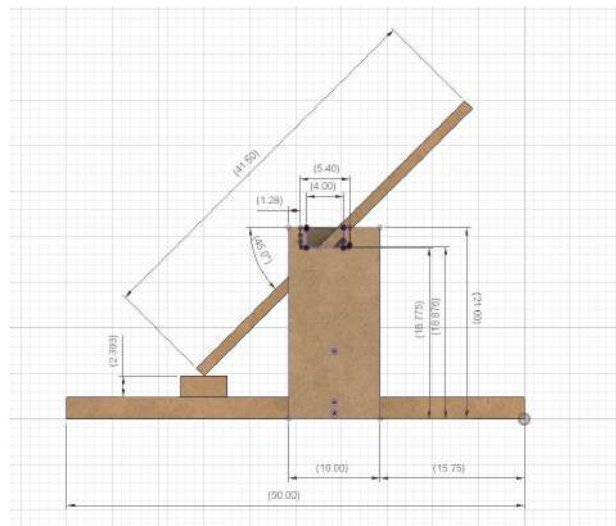


Figura 4 - Vista lateral

Devido ao corpo do equipamento ser fabricado a partir de uma única placa de MDF com dimensões de 2000mm x 100mm x 12mm, como representado na figura 5, tornou-se necessário realizar adaptações no acoplamento do mancal a base, permitindo assim acoplar o servo MG995 sem a necessidade de alterar o dimensionamento da rampa principal.

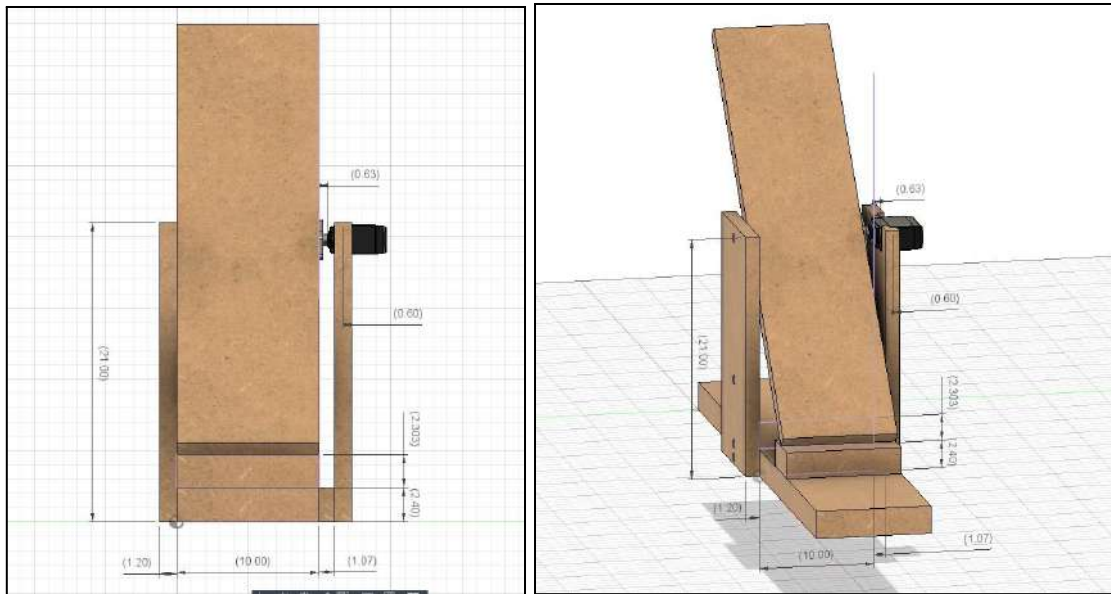


Figura 5 - Vista Frontal e Isométrica

## 2.6 Código

Para permitir o funcionamento automatizado do sistema foram desenvolvidos 2 códigos complementares. o primeiro consiste em um código em C++ para definir os parâmetros de ponto morto e velocidade de rotação da rampa, calibração do giroscópio e comunicação com o arduino.

Apesar da possibilidade de utilizar um servo motor, o qual informa ao usuário a posição exata em um determinado instante, foi necessário utilizar um servo de 360° de rotação devido ao alto peso do corpo que precisa ser movimentado. Ao alterar o servo, o hardware foi adaptado de forma a permitir uma inclinação correta diretamente pela informação fornecida pelo giroscópio.

```
#include <Servo.h>
#include <Wire.h>

Servo meuServo;

// PINAGEM
const int pinoServo = 10;
const int pinoTrig = 7;
const int pinoEcho = 6;

// CALIBRAÇÃO DO SERVO (Rotação Contínua)
const int pontoMorto = 60;
const int velocidadePositiva = 85;
const int velocidadeNegativa = 40;

// Endereço do Giroscópio
const int MPU_ADDR = 0x68;

// Variáveis Globais
float anguloAtualX = 0;
float erroGiroX = 0;
unsigned long tempoAnteriorGiro = 0;
```

```
unsigned long tempoAnteriorSonar = 0;

// SETUP
void setup() {
    Serial.begin(9600);
    Wire.begin();

    // Configuração dos Pinos
    pinMode(pinoTrig, OUTPUT);
    pinMode(pinoEcho, INPUT);
    meuServo.attach(pinoServo);
    meuServo.write(pontoMorto);

    // Inicializa MPU6050
    Serial.println("Iniciando MPU6050...");
    Wire.beginTransmission(MPU_ADDR);
    Wire.write(0x6B);
    Wire.write(0);
    Wire.endTransmission(true);

    // Calibração do Giroscópio
    Serial.println("Calibrando Giroscopio... MANTENHA PARADO!");
    long soma = 0;
    // Precisamos da função lerGiroX_Raw aqui, ela está definida no fim do arquivo
    for(int i = 0; i < 1000; i++) {
        soma += lerGiroX_Raw();
        delay(3);
    }
    erroGiroX = soma / 1000.0;

    // MENSAGEM
    Serial.println("--- Sistema Pronto: Digite o angulo ---");

    tempoAnteriorGiro = millis();
}

// LOOP PRINCIPAL
void loop() {
    // Mantém o ângulo atualizado o tempo todo
    atualizarAnguloX();

    if (Serial.available() > 0) {
        // Lê a string inteira até o 'Enter' para limpar o buffer corretamente
        String leitura = Serial.readStringUntil('\n');
        leitura.trim(); // Remove espaços extras

        if (leitura.length() > 0) {
            float anguloAlvo = leitura.toFloat();

            // Se recebeu um ângulo válido (diferente de zero), executa a sequência
            if (anguloAlvo != 0) {
                // 1. Move a rampa
                moverAteAngulo(anguloAlvo);

                // 2. Grava os dados da descida
                executarStreamingDados();
            }
        }
    }
}
```



```
    }  
  }  
}  
}  
  
// 1. Lógica do Giroscópio (Leitura RAW)  
int16_t lerGiroX_Raw() {  
  Wire.beginTransaction(MPU_ADDR);  
  Wire.write(0x43); // Registrador do Eixo X do Giroscópio  
  Wire.endTransmission(false);  
  Wire.requestFrom(MPU_ADDR, 2, true);  
  
  int16_t eixoX = Wire.read() << 8 | Wire.read();  
  return eixoX;  
}  
  
// 2. Atualização do Ângulo (Integração no tempo)  
void atualizarAnguloX() {  
  unsigned long tempoAtual = millis();  
  float dt = (tempoAtual - tempoAnteriorGiro) / 1000.0;  
  tempoAnteriorGiro = tempoAtual;  
  float leituraRaw = lerGiroX_Raw();  
  float velocidadeAngular = (leituraRaw - erroGiroX) / 131.0;  
  
  // Zona morta para evitar drift quando parado  
  if (abs(velocidadeAngular) < 1.5) {  
    velocidadeAngular = 0;  
  }  
  anguloAtualX += velocidadeAngular * dt;  
}  
  
// 3. Movimento do Motor  
void moverAteAngulo(float grausDesejados) {  
  anguloAtualX = 0; // Zera para movimento relativo  
  
  Serial.print(">>> Movendo para: ");  
  Serial.println(grausDesejados);  
  
  if (grausDesejados > 0) {  
    meuServo.write(velocidadePositiva);  
  } else {  
    meuServo.write(velocidadeNegativa);  
  }  
  
  bool chegou = false;  
  
  while (!chegou) {  
    atualizarAnguloX();  
  
    if (abs(anguloAtualX) >= abs(grausDesejados)) {  
      chegou = true;  
    }  
  }  
  
  meuServo.write(pontoMorto);  
  delay(1000); // Espera 1s para estabilizar a rampa
```



```
Serial.println(">>> Chegou"); // Avisar o Python
}

// 4. Leitura Ultrassônica Rápida (Para o Streaming)
float lerUltrassomRapido() {
    digitalWrite(pinoTrig, LOW);
    delayMicroseconds(2);
    digitalWrite(pinoTrig, HIGH);
    delayMicroseconds(10);
    digitalWrite(pinoTrig, LOW);

    // Timeout curto (25ms) para garantir alta taxa de quadros (fps)
    long duration = pulseIn(pinoEcho, HIGH, 25000);

    if (duration == 0) return -1.0;
    return duration * 0.034 / 2;
}

// 5. Execução do Streaming (Envia dados para o Python)
void executarStreamingDados() {
    Serial.println("START_STREAM");

    unsigned long tempoInicial = millis();
    unsigned long tempoAtual = 0;

    // Grava por 3 segundos
    while (tempoAtual < 3000) {
        atualizarAnguloX(); // Mantém o gyro vivo em background

        tempoAtual = millis() - tempoInicial;
        float dist = lerUltrassomRapido();

        // Formato CSV: TEMPO,DISTANCIA
        Serial.print(tempoAtual);
        Serial.print(",");
        Serial.println(dist);

        delay(35); // Controla a taxa de amostragem
    }

    Serial.println("END_STREAM");
}
```

O segundo código em python focado na análise dos dados através das equações apresentadas no índice 4.2 e salvamento automático de todas as informações adquiridas em um arquivo CSV local.

Devido a natureza dos dados e as limitações dos módulos foram aplicados filtros de medianas com janelas de 9 dados.

```
import serial
import time
import pandas as pd
import seaborn as sns
```

```
import matplotlib.pyplot as plt
import csv
import os
import numpy as np
import math

# Conexões
PORTA_ARDUINO = 'COM13'
BAUD_RATE = 9600
ARQUIVO_CSV = 'experimento_completo.csv'

# PARÂMETROS DO OBJETO (VARIAR)
MASSA_KG = 0.04 # Ex: 200g = 0.2 kg (Altere para o peso real)
GRAVIDADE = 9.81 # m/s^2
LIXACORPO = 220
LIXAMESA = 80

# Configuração do Filtro
JANELA_FILTRO = 9 # alta devido a aceleração

def conectar_arduino():
    try:
        print(f"🔌 Conectando em {PORTA_ARDUINO}...")
        ser = serial.Serial(PORTA_ARDUINO, BAUD_RATE, timeout=1)
        print("⌚ Aguardando inicialização...")

        start_wait = time.time()
        while True:
            if time.time() - start_wait > 15:
                print("❌ Timeout.")
                return None
            if ser.in_waiting:
                linha = ser.readline().decode('utf-8', errors='ignore').strip()
                if "Digite o angulo" in linha or "Sistema Pronto" in linha:
                    print("✅ Arduino Pronto!")
                    break
        return ser
    except Exception as e:
        print(f"❌ Erro: {e}")
        return None

def realizar_teste_streaming(arduino):
    print("\n--- 🏗️ TESTE DE TRIBOMETRO ---")
    try:
        angulo = float(input("📐 Ângulo da Rampa (graus): "))
    except ValueError:
        return

    print(f"📦 Inclinando para {angulo}°...")
    arduino.reset_input_buffer()
    arduino.write(f"{angulo}\n".encode())

    # Aguarda Motor
    while True:
        try:
            linha = arduino.readline().decode('utf-8', errors='ignore').strip()
```

```
        if "Chegou" in linha:
            print("✅ Rampa posicionada. Solte o objeto!")
            break
        except: break

# Captura Dados
dados_tempo = []
dados_distancia = []
gravando = False
start_stream_wait = time.time()

while True:
    if (time.time() - start_stream_wait) > 10: break

    if arduino.in_waiting:
        linha = arduino.readline().decode('utf-8', errors='ignore').strip()

        if "START_STREAM" in linha:
            print("🔴 Gravando...")
            gravando = True
            continue
        if "END_STREAM" in linha:
            print("🟩 Fim da gravação.")
            break

        if gravando and "," in linha:
            try:
                p = linha.split(',')
                t_s = int(p[0]) / 1000.0
                d_cm = float(p[1])
                if 0 < d_cm < 400:
                    dados_tempo.append(t_s)
                    dados_distancia.append(d_cm)
            except: pass

    if len(dados_tempo) > 10:
        processar_fisica(dados_tempo, dados_distancia, angulo)
    else:
        print("⚠️ Dados insuficientes.")

def processar_fisica(tempo, distancias, angulo_graus):
    """Realiza todos os cálculos físicos e plota 4 gráficos"""

    # 1. Preparar Dados no Pandas
    df = pd.DataFrame({'t': tempo, 's_cm': distancias})

    # 2. Filtragem (Crucial para derivadas não explodirem)
    df['s_filt'] = df['s_cm'].rolling(window=JANELA_FILTRO, center=True).median()
    df['s_filt'] = df['s_filt'].fillna(df['s_cm']) # Preenche bordas

    # Converter para Sistema Internacional (Metros)
    # O sensor está no TOPO e mede a distância aumentando
    # S = posição na rampa.
    df['s_m'] = df['s_filt'] / 100.0

    # 3. CÁLCULOS CINEMÁTICOS (Derivadas Numéricas)
```

```
# dt é o intervalo de tempo entre medidas
dt = np.gradient(df['t'])

# Velocidade (v = ds/dt)
df['v'] = np.gradient(df['s_m'], df['t'])

# Aceleração (a = dv/dt)
# Aceleração costuma ser muito ruidosa, aplicamos um filtro extra nela ou na
velocidade
df['v_smooth'] = df['v'].rolling(window=5, center=True).mean().fillna(df['v'])
df['a'] = np.gradient(df['v_smooth'], df['t'])

# 4. CÁLCULOS DINÂMICOS & ENERGIA
theta_rad = math.radians(angulo_graus)
sen_t = math.sin(theta_rad)
cos_t = math.cos(theta_rad)

# Altura (h): d=0 é o topo. h diminui conforme d aumenta.
# Definimos h = 0 no ponto final da medição para facilitar referência
dist_max = df['s_m'].max()
df['h'] = (dist_max - df['s_m']) * sen_t

# Energias (Joules)
df['U'] = MASSA_KG * GRAVIDADE * df['h']          # Potencial Gravitacional
df['K'] = 0.5 * MASSA_KG * (df['v']**2)          # Cinética
df['E_mec'] = df['U'] + df['K']                  # Mecânica Total

# Perda de Energia (Acumulada)
# Quanto de energia "sumiu" desde o início do movimento
E_inicial = df['E_mec'].iloc[0] # Ou a média dos primeiros pontos
df['Perda_Energia'] = E_inicial - df['E_mec']

# 5. CÁLCULO DO ATRITO DINÂMICO (Instantâneo)
#  $\mu = \tan(\theta) - a / (g * \cos(\theta))$ 
# Evita divisão por zero e valores absurdos gerados por ruído no início
df['mu_d'] = math.tan(theta_rad) - (df['a'] / (GRAVIDADE * cos_t))

# Limpeza de ruídos matemáticos no coeficiente (filtro de saturação)
df['mu_d'] = df['mu_d'].clip(lower=0, upper=1.0)

# 6. TRABALHO DA FORÇA DE ATRITO
# Força Normal  $N = m * g * \cos(\theta)$ 
#  $Fat = \mu * N$ 
# Trabalho = Integral de  $Fat * ds$ 
# Como  $\mu$  varia (devido ao ruído), calculamos o trabalho incremental  $dW = Fat * ds$ 

Normal = MASSA_KG * GRAVIDADE * cos_t
df['Fat_inst'] = df['mu_d'] * Normal

# Deslocamento incremental (ds)
ds = np.gradient(df['s_m'])
df['Trab_Fat_Inst'] = df['Fat_inst'] * ds
df['Trab_Fat_Acumulado'] = df['Trab_Fat_Inst'].cumsum()

# PLOTAGEM DOS 4 GRÁFICOS
```

```
fig, axs = plt.subplots(2, 2, figsize=(14, 10))
fig.suptitle(f'Análise Física Completa - Rampa {angulo_graus}°', fontsize=16)

# Gráfico 1: Trajetória e Velocidade
ax1 = axs[0, 0]
ax1.plot(df['t'], df['s_m'], label='Posição (m)', color='blue')
ax1b = ax1.twinx() # Eixo secundário
ax1b.plot(df['t'], df['v'], label='Velocidade (m/s)', color='orange',
linestyle='--')
ax1.set_title('Cinemática')
ax1.set_xlabel('Tempo (s)')
ax1.set_ylabel('Posição (m)')
ax1b.set_ylabel('Velocidade (m/s)')
ax1.legend(loc='upper left')
ax1b.legend(loc='lower right')

# Gráfico 2: Conservação de Energia
ax2 = axs[0, 1]
ax2.plot(df['t'], df['U'], label='Potencial (U)', color='green')
ax2.plot(df['t'], df['K'], label='Cinética (K)', color='red')
ax2.plot(df['t'], df['E_mec'], label='Mecânica Total (E)', color='black',
linewidth=2)
ax2.set_title('Energias (Joules)')
ax2.set_xlabel('Tempo (s)')
ax2.set_ylabel('Energia (J)')
ax2.legend()
ax2.grid(True, linestyle='--', alpha=0.5)

# Gráfico 3: Coeficiente de Atrito Dinâmico
ax3 = axs[1, 0]
# Filtramos para mostrar apenas quando há movimento real (v > 0.05)
mask_mov = df['v'] > 0.05
ax3.plot(df.loc[mask_mov, 't'], df.loc[mask_mov, 'mu_d'], color='purple',
alpha=0.6)
# Linha média
mu_medio = df.loc[mask_mov, 'mu_d'].mean()
ax3.axhline(mu_medio, color='black', linestyle='--', label=f'Média =
{mu_medio:.3f}')
ax3.set_title('Coeficiente de Atrito Dinâmico Estimado')
ax3.set_xlabel('Tempo (s)')
ax3.set_ylabel('μ Dinâmico')
ax3.set_ylim(0, 1.0) # Limita eixo Y para não distorcer com ruídos
ax3.legend()

# Gráfico 4: Comparação Trabalho vs Perda
ax4 = axs[1, 1]
ax4.plot(df['t'], df['Perda_Energia'], label='Perda de Energia Mecânica (ΔE)',
color='red')
ax4.plot(df['t'], df['Trab_Fat_Acumulado'], label='Trabalho do Atrito (W_fat)',
color='blue', linestyle='--')
ax4.set_title('Trabalho Não-Conservativo vs Perda Energética')
ax4.set_xlabel('Tempo (s)')
ax4.set_ylabel('Energia / Trabalho (J)')
ax4.fill_between(df['t'], df['Perda_Energia'], df['Trab_Fat_Acumulado'],
color='gray', alpha=0.2, label='Erro Experimental')
ax4.legend()
```

```
ax4.grid(True)

plt.tight_layout()
plt.show()

# Salva CSV Detalhado
df.to_csv(f"analise_fisica_{angulo_graus}.csv", index=False)
print(f"💾 Dados detalhados salvos em
'analise_fisica_{angulo_graus}GRAUS_{MASSA_KG}KG_C_{LIXACORPO}_L_{LIXAMESA}.csv'")
print(f"📁 O arquivo será salvo em:", os.getcwd())
def main():
    arduino = conectar_arduino()
    if not arduino: return
    while True:
        opt = input("\n[Enter] Novo Teste | [S] Sair: ")
        if opt.lower() == 's': break
        realizar_teste_streaming(arduino)
    arduino.close()

if __name__ == "__main__":
    main()
```

### 3. Fabricação

O processo de fabricação seguiu etapas fundamentais para garantir a eficiência e redução de custos, combinando com uma prioridade em materiais estruturais baratos, acessíveis e sustentáveis, além da utilização de componentes eletrônicos de baixo custo.

#### 3.1 Montagem Estrutural e Mecânica

A base do tribômetro foi construída utilizando chapas de MDF de 6mm. A estrutura foi dimensionada para garantir rigidez durante a elevação da rampa, e suas dimensões foram feitas pensando em pré-requisitos, como a segurança e a garantia de estabilidade.

Inicialmente as chapas foram coladas e cortadas, seguindo o projeto CAD e as dimensões que queríamos para esse modelo, como é visto na figura 6 e 7.



figura 6 e 7- momento do corte das chapas e peças de MDF já finalizadas

Seguindo o modelo 3D para validação de encaixes e otimização da manufatura antes do corte, a montagem continuou com a fixação das peças de MDF usando parafusos e cola, assim finalizando a parte estrutural.

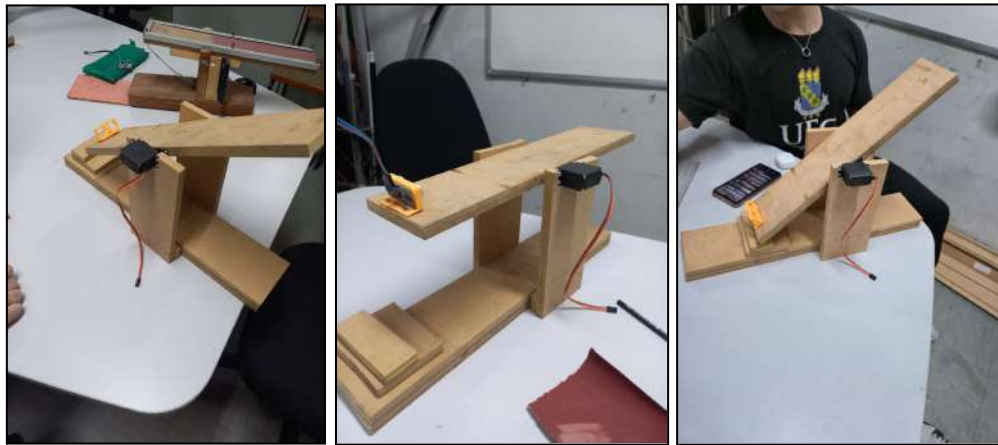


Figura 8, 9, 10 - Vistas isométricas do modelo estrutural finalizado

Também foi feito uma case de suporte para o sensor de ultrassom, localizado na parte posterior da rampa para evitar interferência no movimento do corpo de prova ou impactos diretos no sensor de ultrassom, essa case também foi fixado com parafusos na rampa, garantindo um apoio seguro e a garantia que o sensor esteja bem acoplado eliminando erros de leitura causados por vibração ou desalinhamento, assim garantindo a fidelidade da sua medição.



Figura 11, 12 e 13 - Case do sensor de ultrassom fixado

Como visto na figura 8, o servo motor MG995 foi acoplado na rampa e devido ao torque necessário, utilizou-se um parafuso como eixo passante para aliviar a carga radial no eixo do motor.



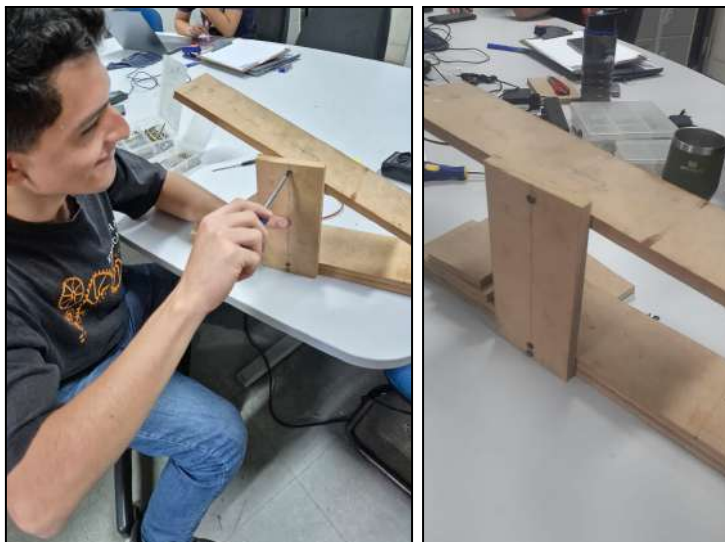


Figura 14 e 15 - Montagem do eixo de apoio e vista isométrica do eixo montado

O projeto do tribômetro finalizado tem a disposição 3 corpos de prova com massa e dimensões diferentes, eles também são de MDF, passando pelo mesmo processo de serem inicialmente cortados e colados, seguindo as dimensões desejadas e uma mudança de massa para as combinações realizadas.



Figura 16 - Corpo de prova

Para a comparação de resultados e análise do impacto da mudança do coeficiente de atrito, foram utilizadas 3 lixas diferentes (P80, P120 e P220) para o corpo de prova e para a rampa. A lixa do corpo de prova foram presas com parafusos

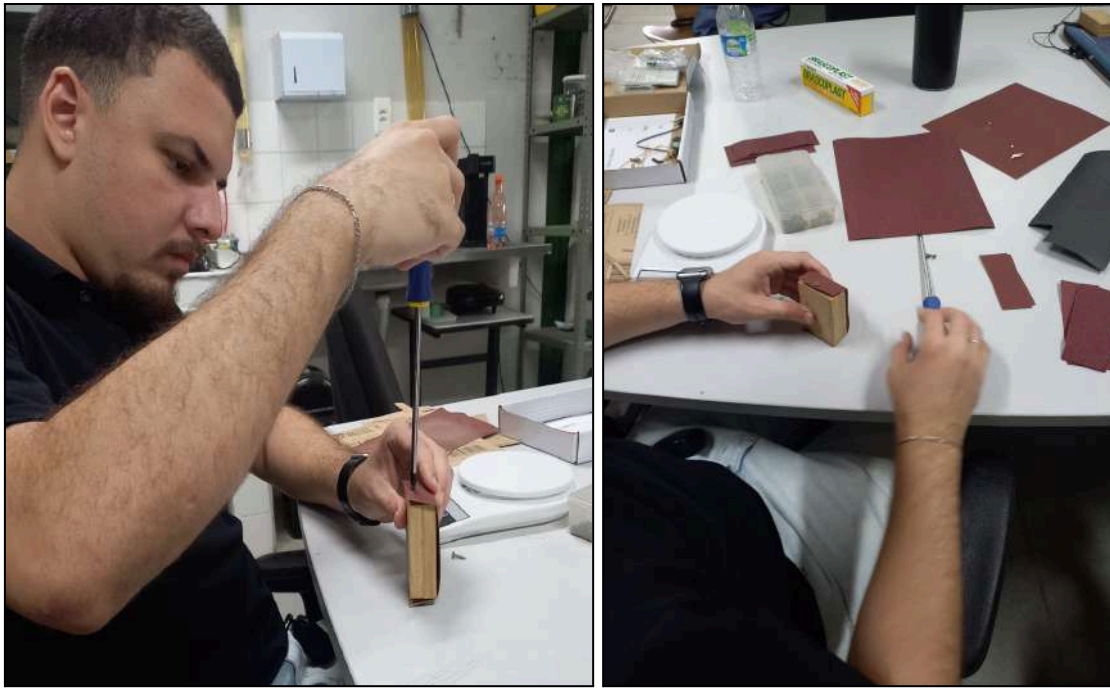


Figura 17 e 18 - fixação das lixas nos corpos de prova

Entretanto, para a lixa da rampa foi utilizado tubos de PVC, que foram cortados seguindo o comprimento da rampa e considerando sua estrutura, de forma que não compromettesse com o movimento da estrutura inclinada.

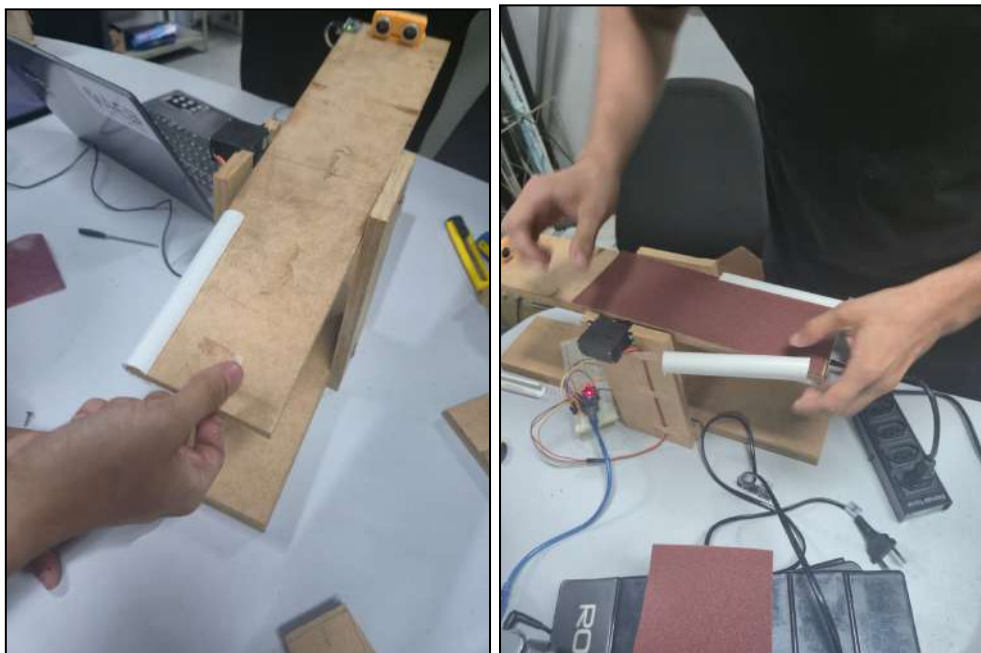


Figura 19 e 20 - instalação dos tubos de PVC na rampa

Logo o modelo finalizado de tribômetro é o que está na figura 21 e 22, na qual sua fabricação seguiu uma montagem simples e acessível, com materiais baratos e fáceis de encontrar.

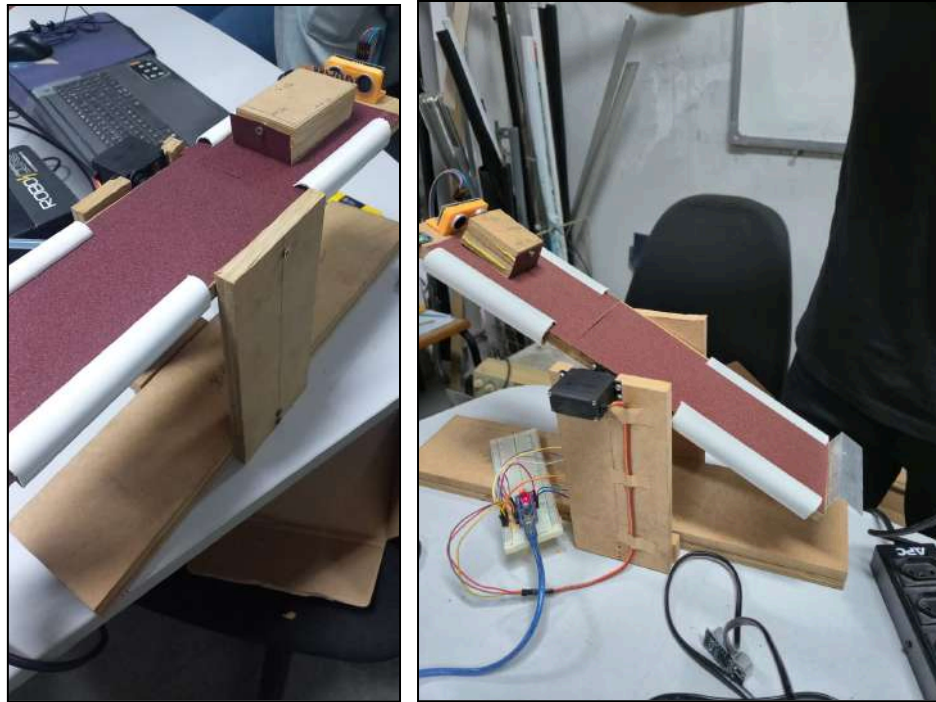


Figura 21 e 22 - Vista isométrica do projeto finalizado

#### 4. Material utilizado e total gasto:

Para realização do trabalho foi utilizado muitos materiais que já estavam à nossa disposição, sejam eles de propriedade dos integrantes ou de trabalhos acadêmicos realizados por nós, a exemplo o MDF de 6mm que foi reutilizado de um trabalho antigo da disciplina de dinâmica dos sistemas mecânicos. O MDF foi utilizado como elemento estrutural do nosso tribômetro, com ele foi possível fazer a base, pista para o deslizamento e corpos de prova. Arduino nano para controle dos sinais. Acelerômetro para nos indicar o ângulo de inclinação da pista, cabos jumpers para conexão dos equipamentos. Motor para movimentar a rampa de deslizamento. A protoboard permite a montagem e teste temporário e sem solda de circuitos eletrônicos. Cola quente para fixar de maneira rápida e fácil o acelerômetro e o suporte do sensor de ultrassom. Parafusos para fixar a base com as partes laterais e pista de ensaio. Suporte impresso em impressora 3D para fixar o sensor de ultrassom na pista de deslizamento. Sensor de ultrassom para adquirir valores de distância percorrida em um determinado tempo. Cola de madeira para colar os corpos de prova para variar as suas massas. E, por fim, parafusos para fixar a lixa nos corpos de prova.

Material	Valor
MDF 6mm	R\$ 30,00
Arduino Nano	R\$ 27,90
Acelerômetro	R\$ 26,50
Cabos Jumps	R\$ 6,57
Motor	R\$ 14,70
Lixas	R\$ 8,00
Protoboard	R\$ 6,56
Cola Quente	R\$ 14,99
Parafusos	R\$ 1,00

Impressão 3D do suporte	R\$ 5,00
Sensor de ultrassom	R\$ 9,40
Cola de madeira	R\$ 9,50
Total	R\$ 160,12

Tabela 1 - Custo de montagem

## 5. Procedimentos Experimentais

Conforme a "Tabela de Possibilidades de Teste" exigida, foram realizados 27 ensaios distintos, variando a massa do corpo de prova e a rugosidade das superfícies, usados para garantir a confiabilidade estatística dos

### dados.5.1 Corpos de Prova e Superfícies

Foram preparados três blocos de madeira com massas aferidas em balança de precisão:

- **Massa 1:** 0,04 kg (leve)
- **Massa 2:** 0,082 kg (média)
- **Massa 3:** 0,103 kg (pesada)

As lixas utilizadas foram classificadas pela sua granulometria:

- **Fina (LBT 1):** P220
- **Média (LBT 2):** P120
- **Grossa (LBT 3):** P80

Portanto a tabela de combinações utilizadas nos ensaios seguiu sendo:

Possibilidade	Massa	LBC	LBT
1	0,04 kg	P220	P220
2	0,04 kg	P220	P120
3	0,04 kg	P220	P80
4	0,04 kg	P120	P220
5	0,04 kg	P120	P120
6	0,04 kg	P120	P80
7	0,04 kg	P80	P220
8	0,04 kg	P80	P120
9	0,04 kg	P80	P80
10	0,082 kg	P220	P220

11	0,082 kg	P220	P120
12	0,082 kg	P220	P80
13	0,082 kg	P120	P220
14	0,082 kg	P120	P120
15	0,082 kg	P120	P80
16	0,082 kg	P80	P220
17	0,082 kg	P80	P120
18	0,082 kg	P80	P80
19	0,103 kg	P220	P220
20	0,103 kg	P220	P120
21	0,103 kg	P220	P80
22	0,103 kg	P120	P220
23	0,103 kg	P120	P120
24	0,103 kg	P120	P80
25	0,103 kg	P80	P220
26	0,103 kg	P80	P120
27	0,103 kg	P80	P80

Tabela 2 - Tabela de possibilidades de teste

**Legenda:** LBC - Lixa da Base do Corpo de Prova; LBT - Lixa da Base de Escorregamento do tribômetro.

## 5.2 Cálculos realizados

### 5.2.1 Dados iniciais e variáveis de entrada

Para a realização dos cálculos, foram definidos os parâmetros físicos fixos do sistema e as variáveis de controle da matriz de testes, na qual essas três variáveis alimentam o restante do código e estão presentes nos cálculos matemáticos.

- **Massa (m):** Encontrada a partir de uma balança digital para cada corpo de prova.
- **Aceleração da Gravidade (g):** 9,81 m/s<sup>2</sup>.
- **Ângulo da Rampa ( $\theta$ ):** A leitura da inclinação é feita em tempo real pelo acelerômetro (MPU6050) fixado na rampa móvel. O sensor atua como



inclinômetro digital, eliminando erros mecânicos e movendo a rampa para a inclinação indicada.

- **Rugosidade das Lixas (Granulometria):** Os testes seguem a tabela de lista de possibilidades.
- **Tempo de Descida (t):** O intervalo de tempo é monitorado pelo sensor ultrassônico, correspondendo à diferença entre o instante da detecção do movimento e a parada do corpo.

### 5.2.2 Determinação do Ângulo de Inclinação ( $\theta$ )

O código do microcontrolador não mede o ângulo diretamente, mas sim os componentes da aceleração gravitacional nos eixos do sensor MPU6050. O ângulo de inclinação da rampa em relação à horizontal é calculado pela função trigonométrica arco-tangente:

$$\theta_x = \arctan\left(\frac{a_y}{a_z}\right) \cdot \left(\frac{180}{\pi}\right)$$

Na qual  $a_y$  é a componente da aceleração no eixo Y (longitudinal da rampa), enquanto  $a_z$  é a componente da aceleração no eixo Z (perpendicular à rampa).

### 5.2.3 Cálculo do Coeficiente de Atrito Estático ( $\mu_s$ )

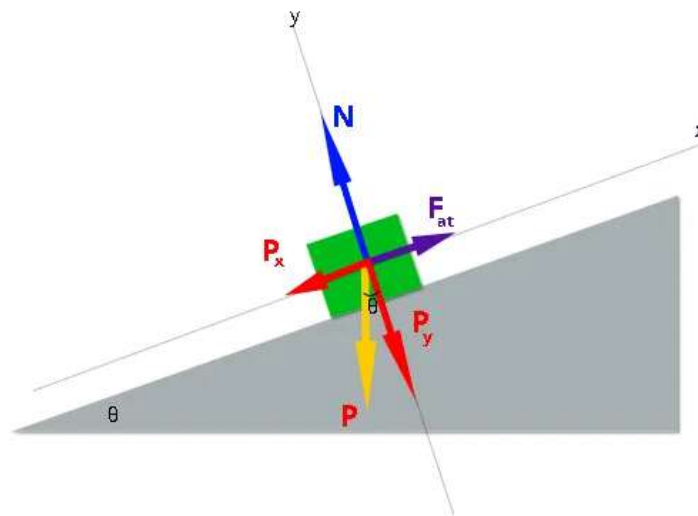


Figura 23 - Diagrama de plano inclinado com atrito

O Cálculo do coeficiente de atrito estático baseado na decomposição vetorial das forças no plano inclinado, na iminência do movimento como visto na figura 23, a componente tangencial do peso ( $P_x$ ) iguala-se à força de atrito estático ( $F_{at}$ ).

$$P_x = m \cdot g \cdot \sin(\theta_x)$$

$$N = m \cdot g \cdot \cos(\theta_x)$$

$$F_{at} = \mu \cdot N$$

Igualando as forças, obtemos a equação simplificada utilizada no software para o cálculo direto:

$$\mu_s = \tan(\theta_x)$$

#### 5.2.4 Cálculo do Coeficiente de Atrito Dinâmico ( $\mu_d$ )

Durante o deslizamento, o corpo está sujeito a uma aceleração resultante. Aplicando a Segunda Lei de Newton no eixo do movimento, como visto na figura 23, na qual:

$$P_x - F_{at} = m \cdot a$$

Portanto substituindo a equação anterior com as equações já formuladas, temos:

$$(m \cdot g \cdot \sin(\theta_x)) - (\mu \cdot N) = m \cdot a$$

Isolando o coeficiente dinâmico, o algoritmo aplica a seguinte fórmula:

$$\mu_d = \tan(\theta_x) - \left( \frac{a}{(g \cdot \cos(\theta_x))} \right)$$

A aceleração média ( $a$ ) é obtida derivando-se a posição fornecida pelo sensor ultrassônico em função do tempo, utilizando a equação do movimento uniformemente variado (MUV), partindo do repouso:

$$a = \frac{2 \cdot d}{t^2}$$

#### 5.2.5 Análise Energética (trabalho energia)

Conforme exigido nas instruções do projeto, realizou-se a validação dos dados através do Teorema Trabalho-Energia. Em um sistema mecânico real, a variação da energia mecânica total deve corresponder ao trabalho realizado pelas forças não-conservativas (neste caso, o atrito).

##### 1. Variação da energia mecânica:

$$W_{Fat} = \Delta E_{mec} = E_f - E_i = E_c - E_p = \frac{1}{2}mv^2 - mg\Delta H$$

##### 2. Trabalho da força de atrito:

$$W_{Fat} = F_{at} \cdot d = \mu_d \cdot N \cdot d = \mu_d \cdot P \cdot \cos(\theta_x) \cdot d$$

O software compara esses dois valores. A coerência do experimento é dada pela proximidade numérica entre  $W_{Fat}$  e  $\Delta E_{mec}$ .

### 6. Resultados experimentais:

Após a conclusão da montagem do protótipo e dos corpos de prova e o posicionamento devido do sensor foram realizados 27 testes para as combinações de granulometrias como pode ser visto na Tabela 2.

Durante cada teste, os dados obtidos pelo sensor ultrassônico foram organizados em 4 gráficos para cada combinação de lixas e corpos de prova, com o objetivo de facilitar a visualização dos resultados das variações de energia mecânica, posição, trabalho e a média experimental do coeficiente de atrito estático



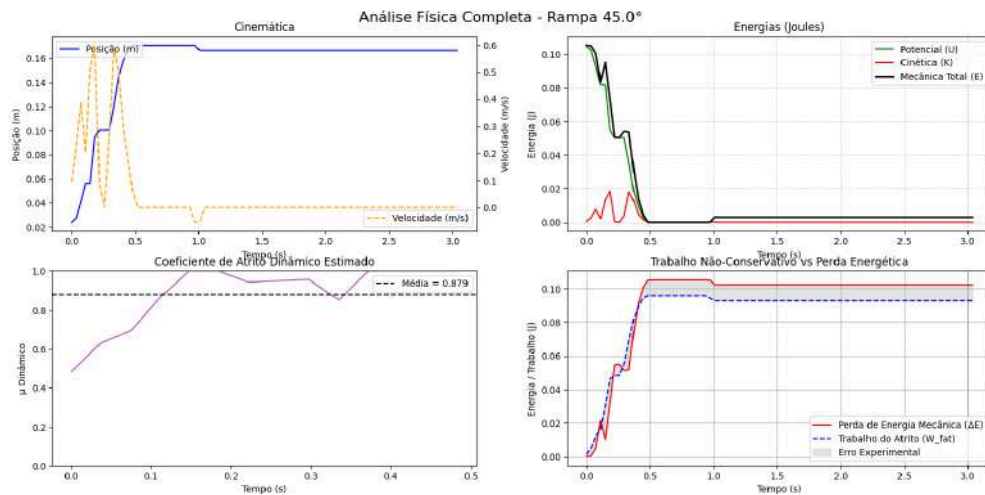


Figura 24 - 103g L220 CP220

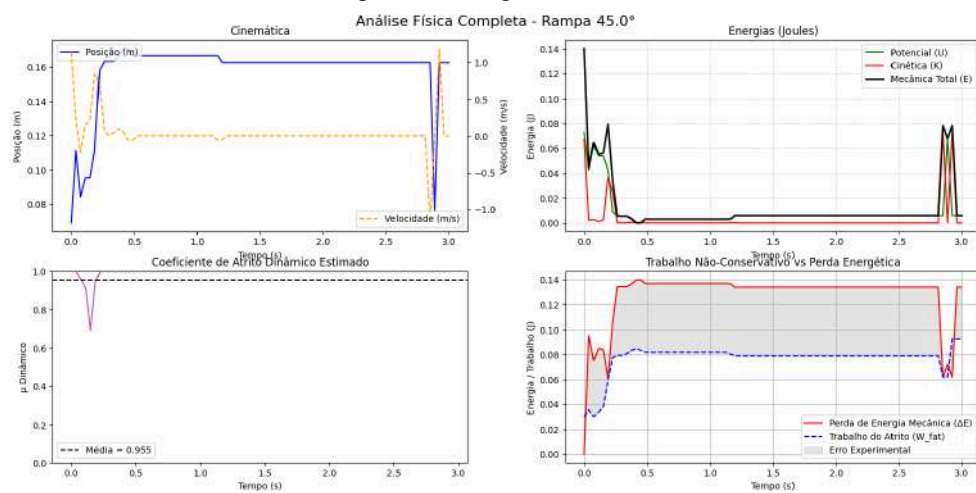


Figura 25 - 103g L220 CP120

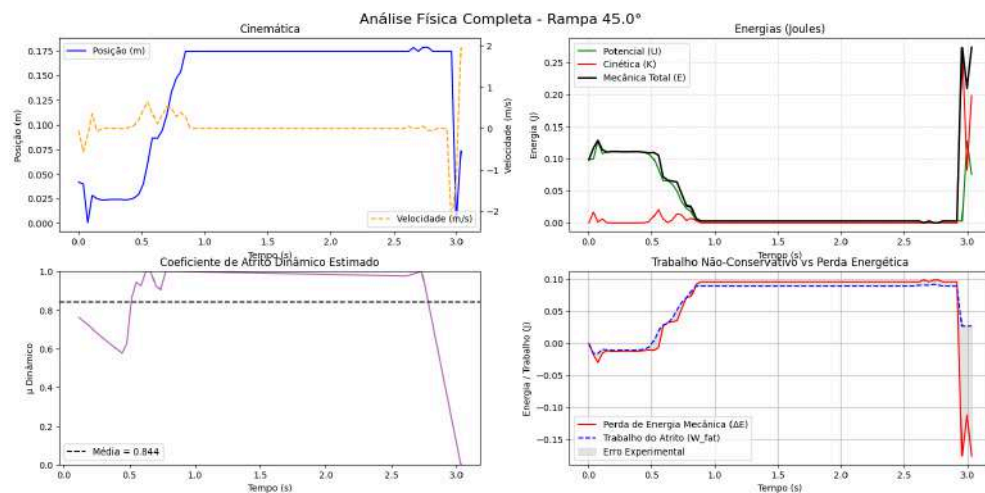


Figura 26 - 103g L220 CP80

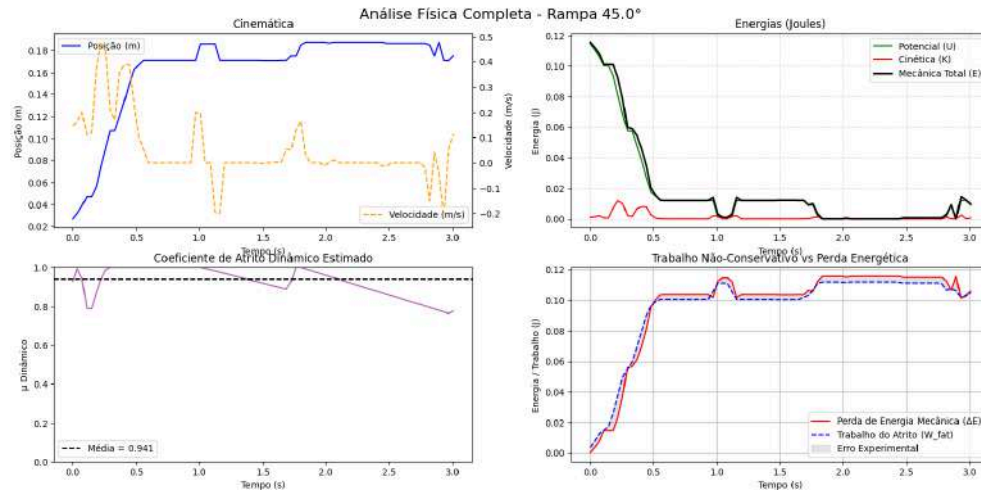


Figura 27 - 103g L120 CP220

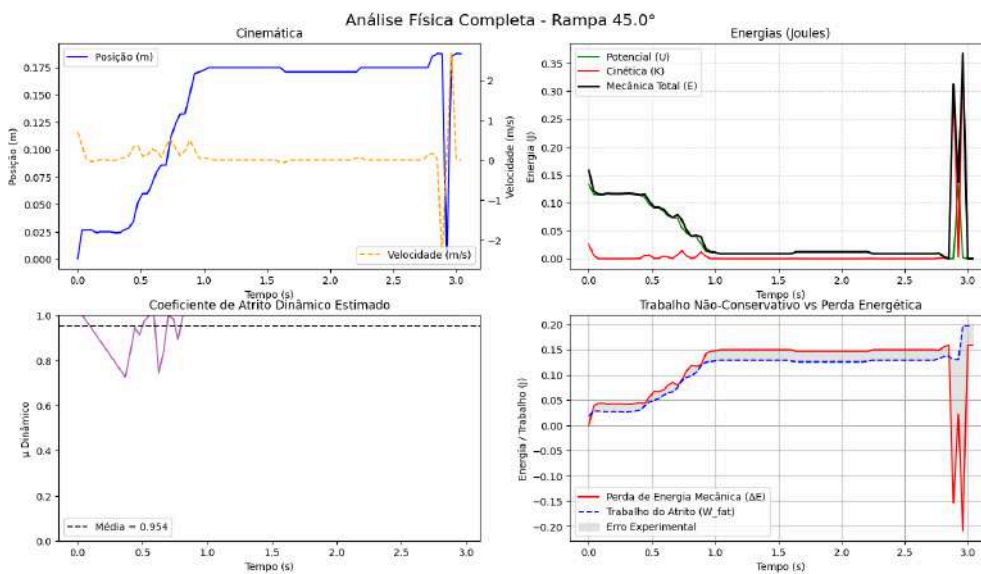


Figura 28 - 103g L120 CP120

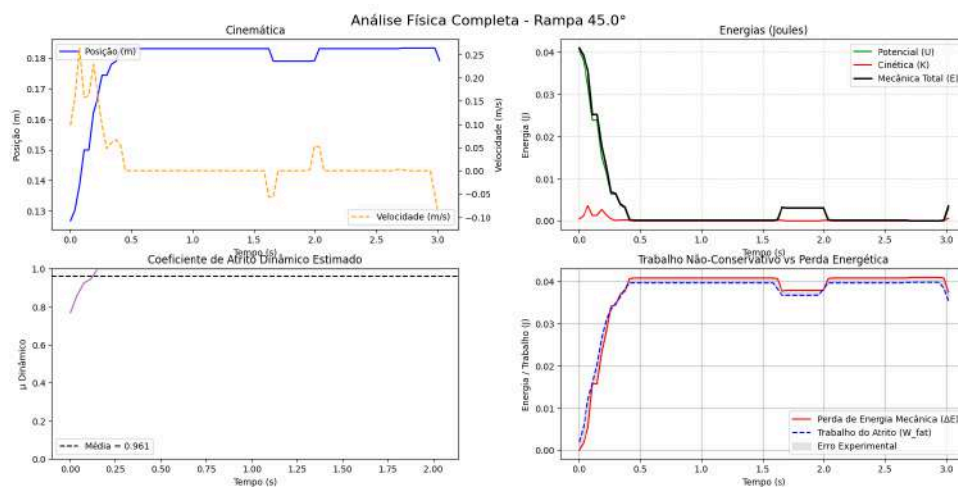


Figura 29 - 103g L120 CP80

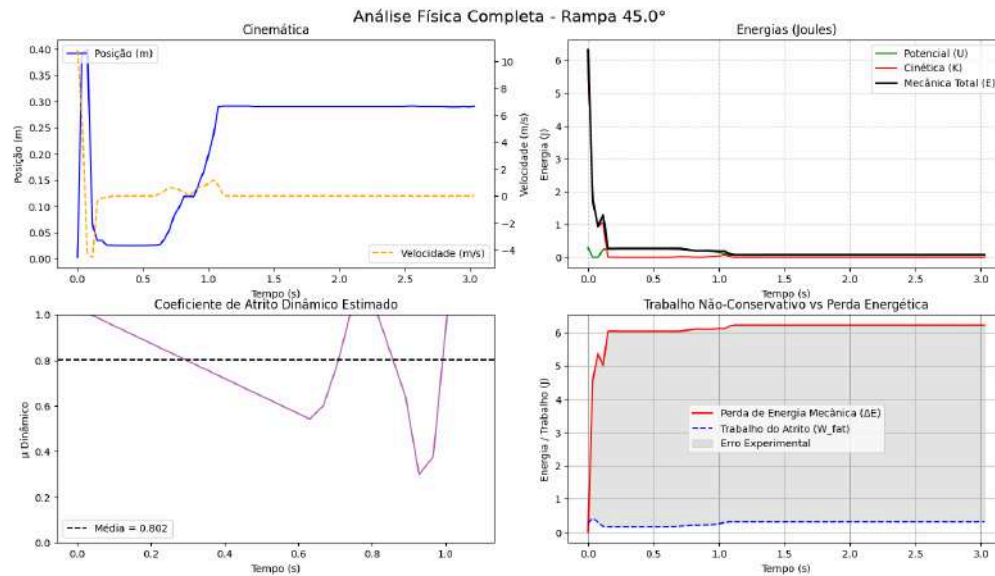


Figura 30 - 103g L80 CP220

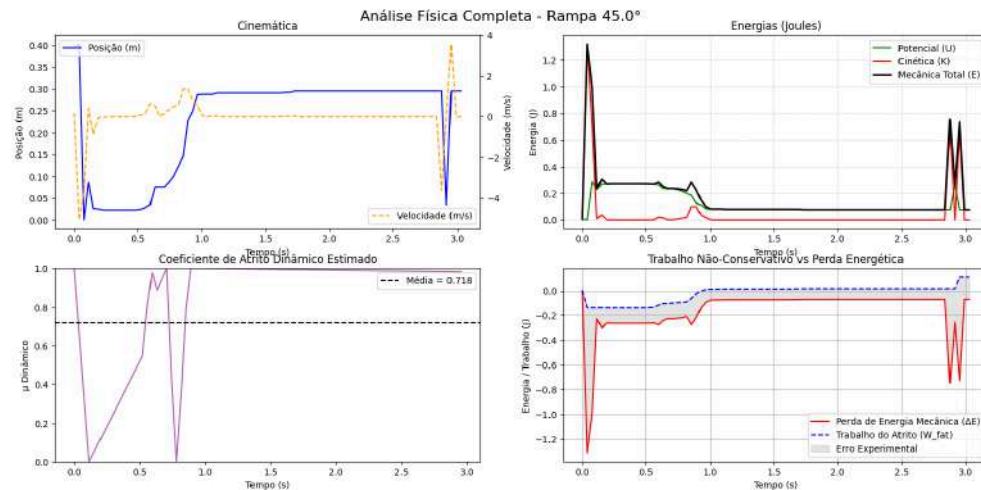


Figura 31 - 103g L80 CP120

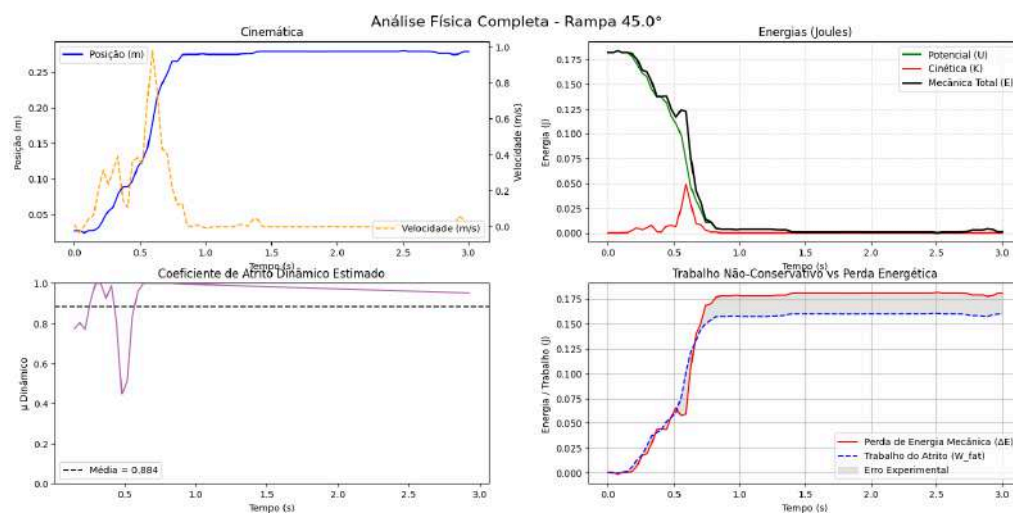


Figura 32 - 103g L80 CP80

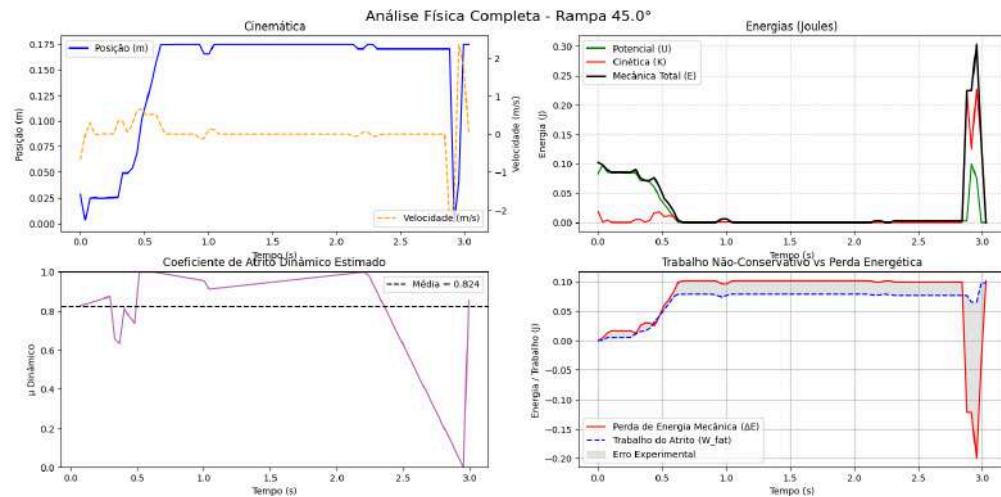


Figura 33 - 82g L220 CP220

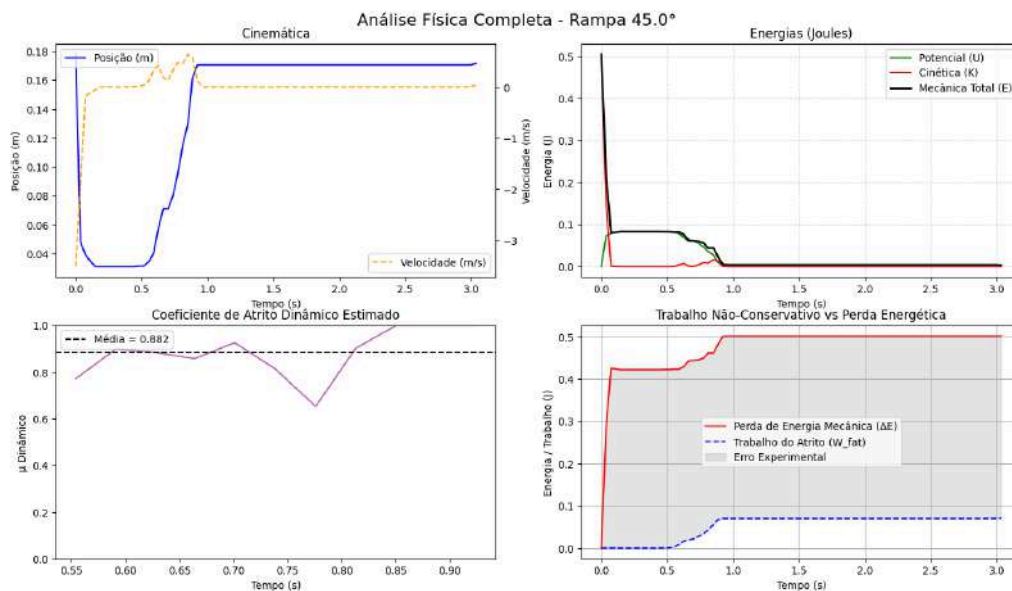


Figura 34 - 82g L220 CP120

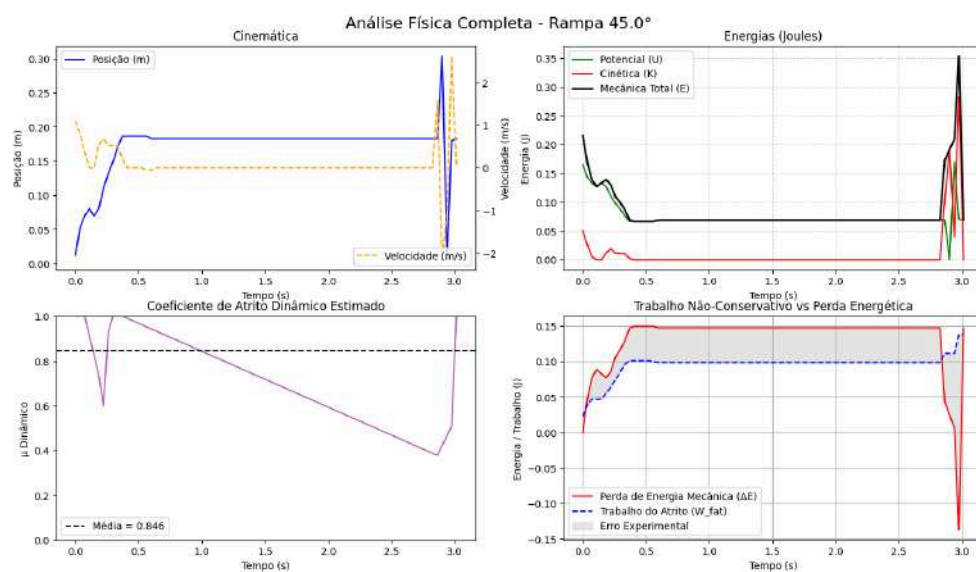




Figura 35 - 82g L220 L80

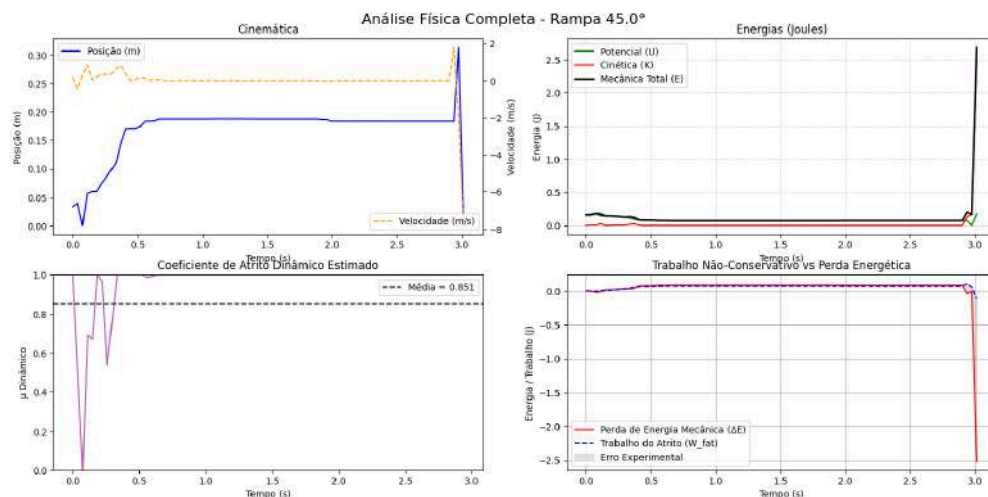


Figura 36 - 82g L120 CP220

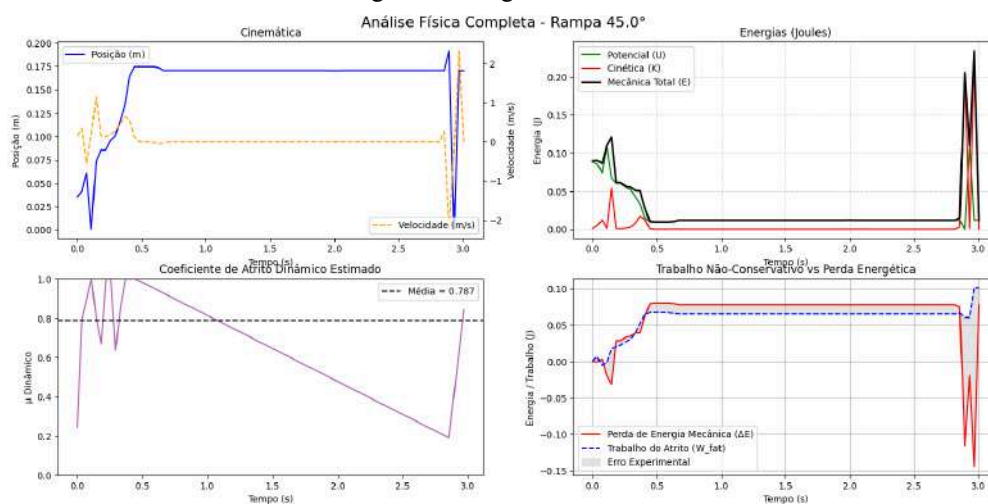


Figura 37 - 82g L120 CP120

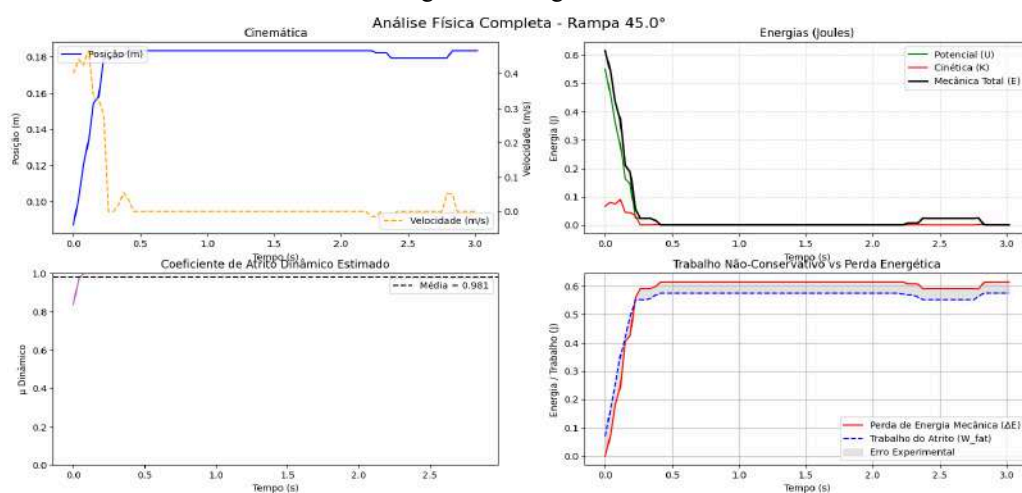


Figura 38 - 82g L120 CP80

Figura

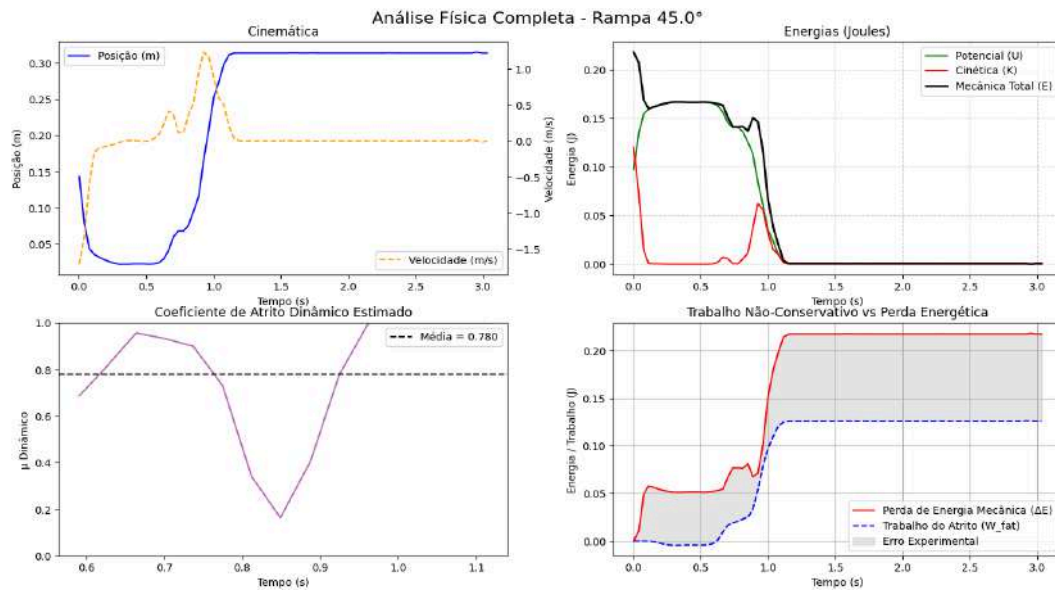


Figura 39 - 82g L80 CP220

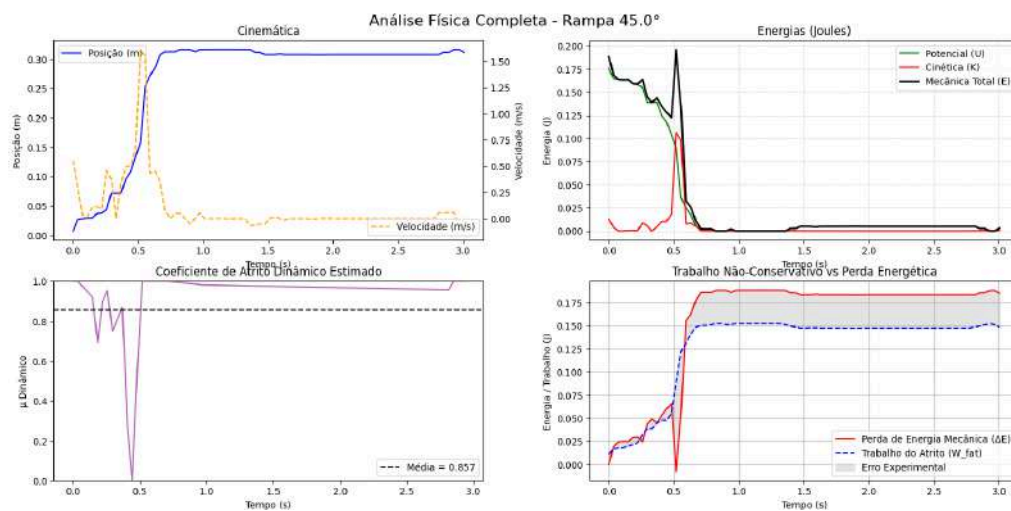


Figura 40 - 82g L80 CP120

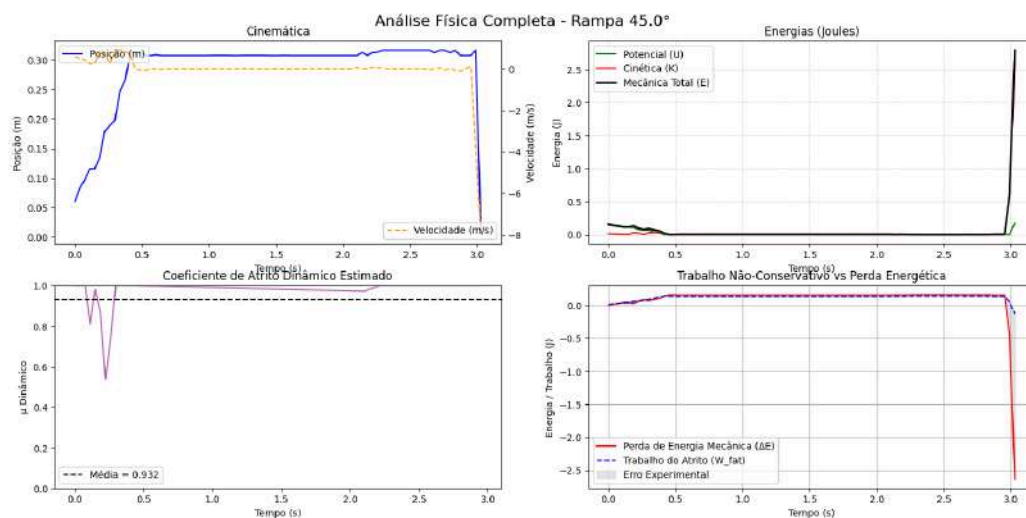


Figura 41 - 82g L80 CP80

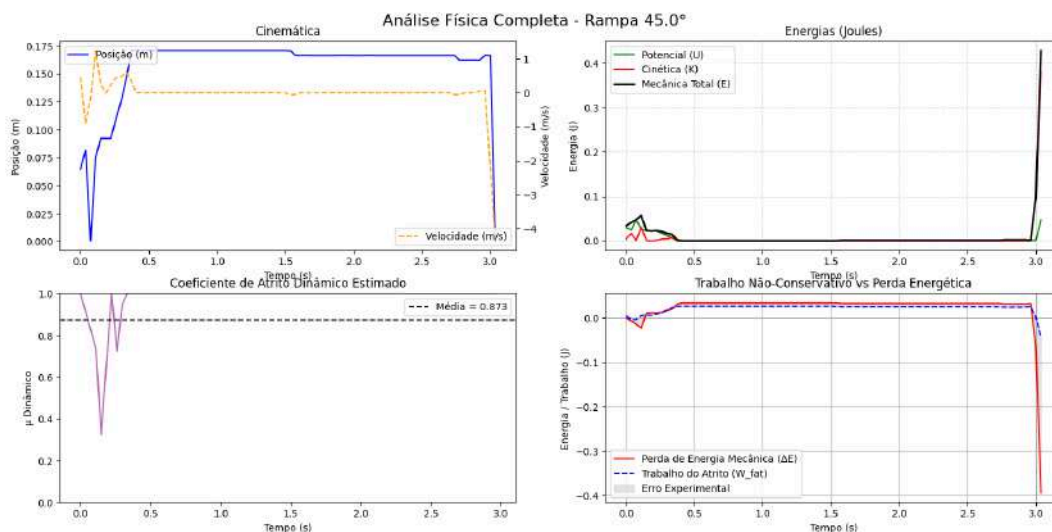


Figura 42 - 40g L220 CP220

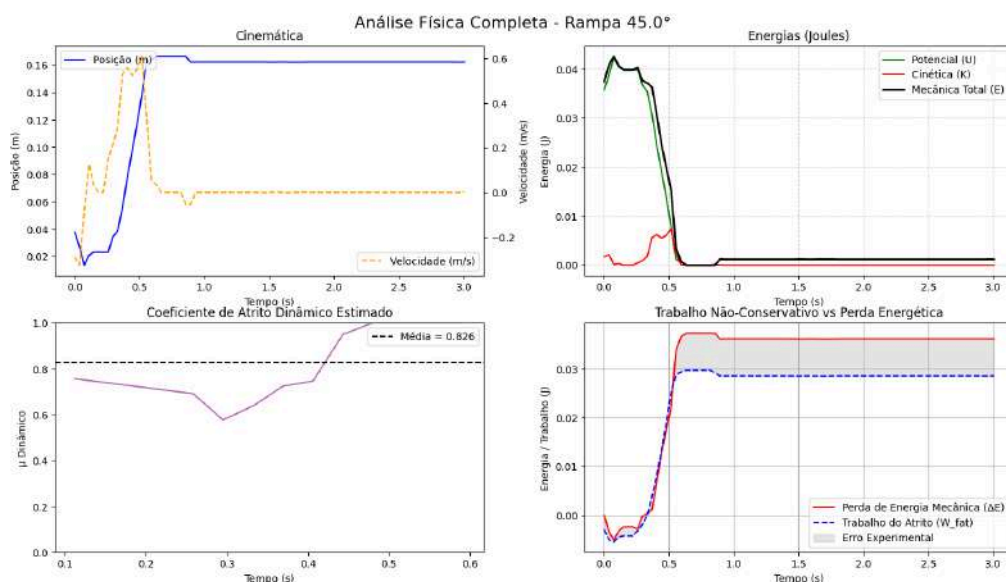


Figura 43 - 40g L220 CP120

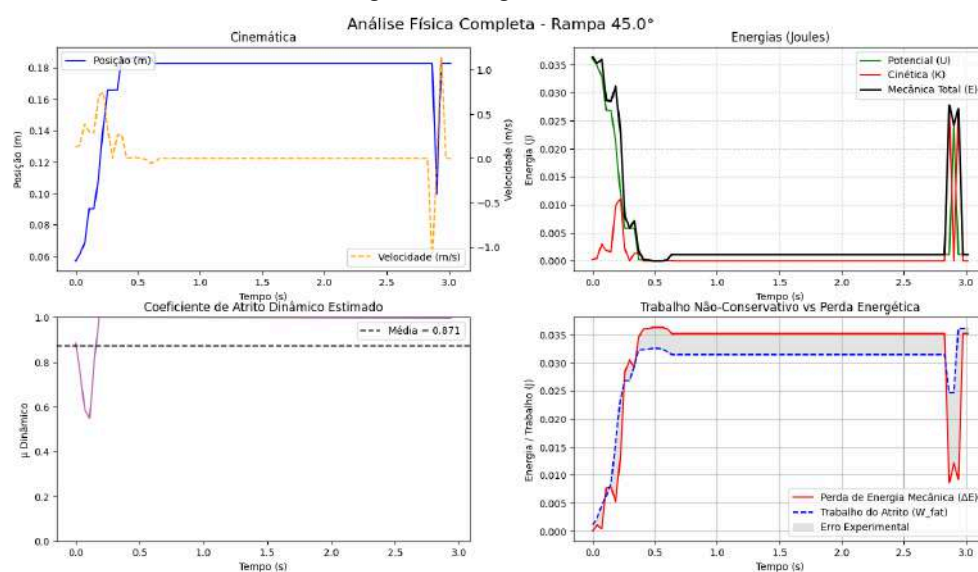


Figura 44 - 40g L220 CP80



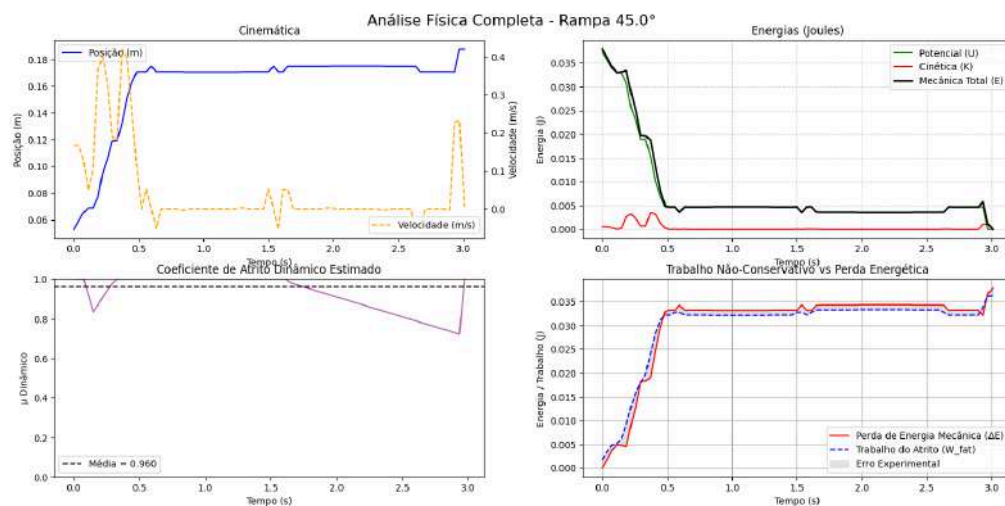


Figura 45 - 40g L120 CP220

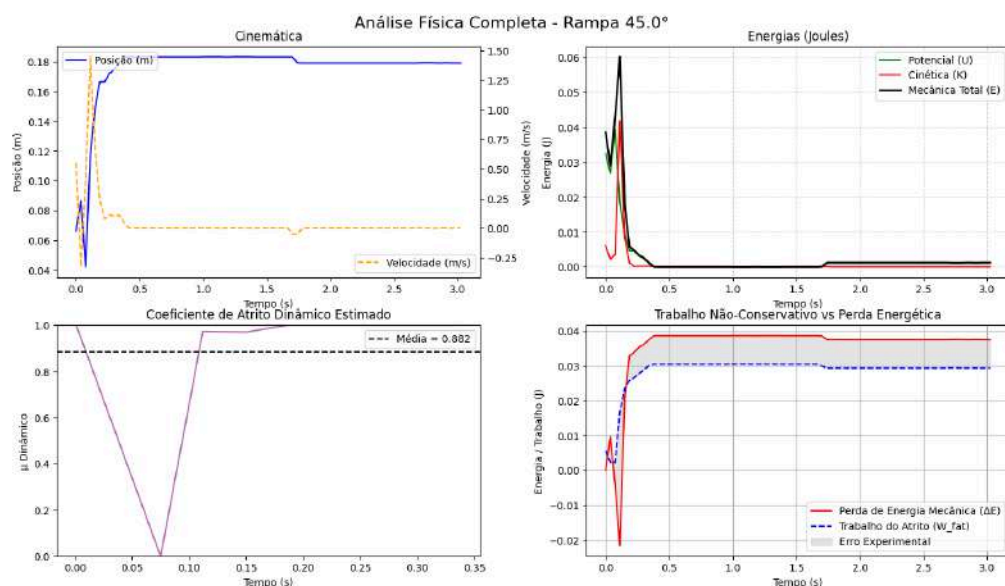


Figura 46 - 40g L120 CP120

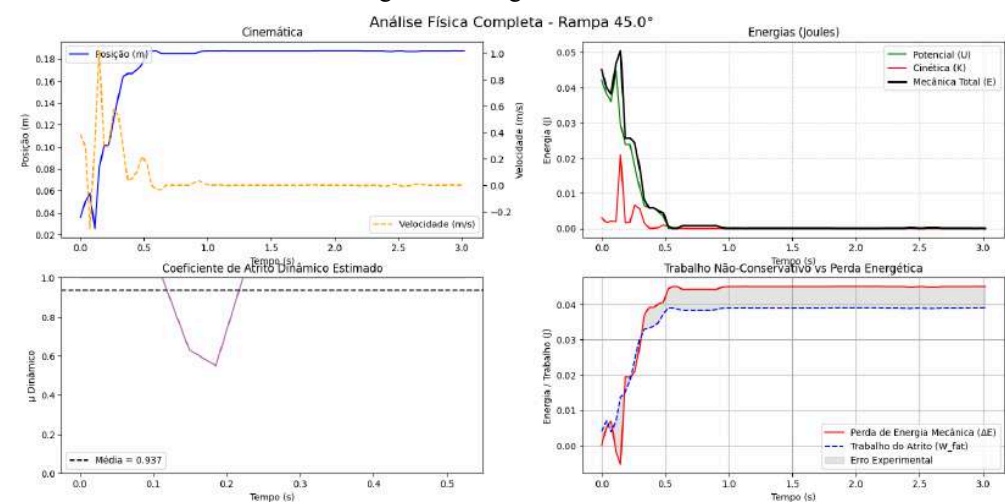


Figura 47 - 40g L120 CP80

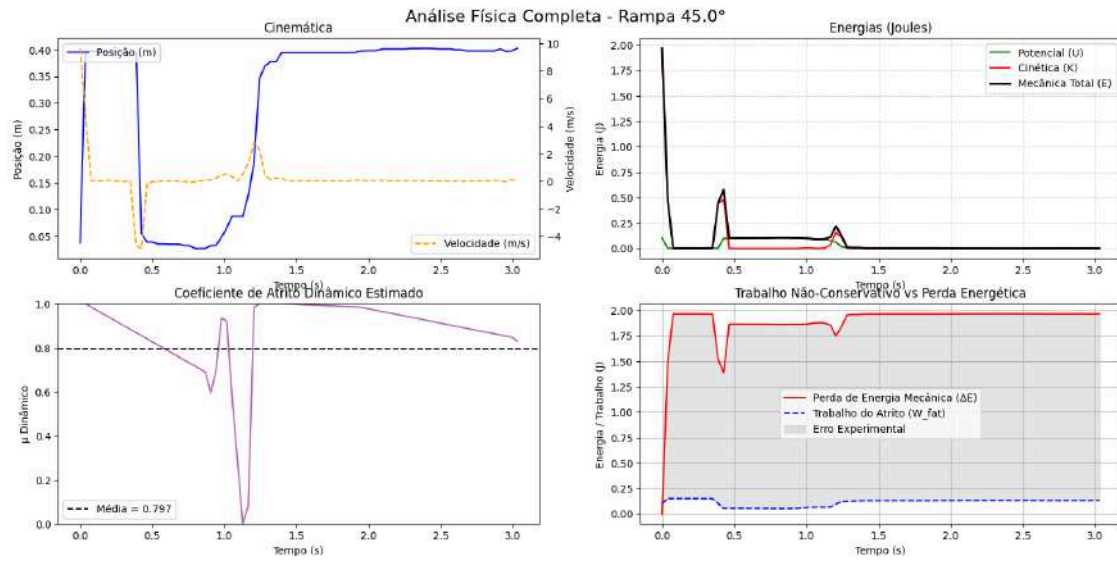


Figura 48 - 40g L80 CP220

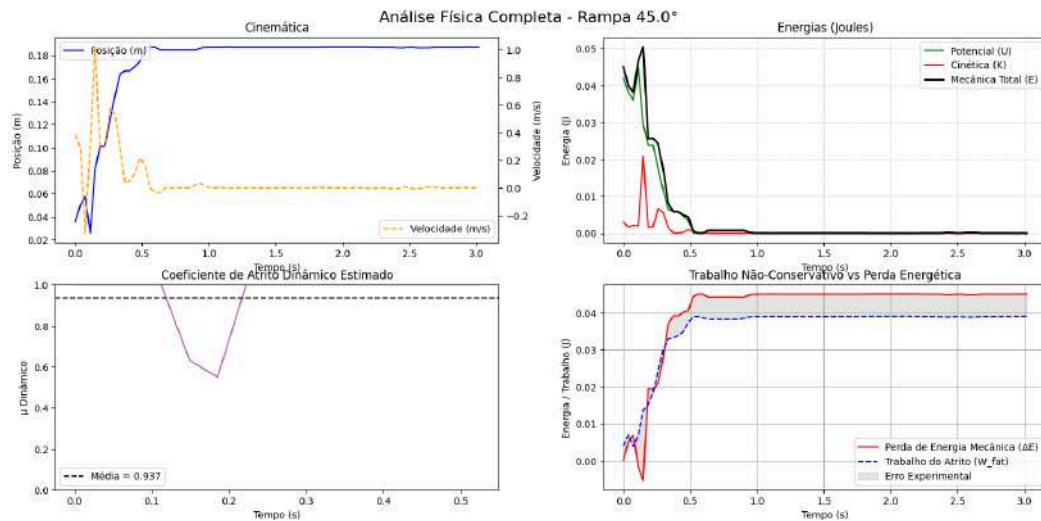


Figura 49 - 40g L80 CP120

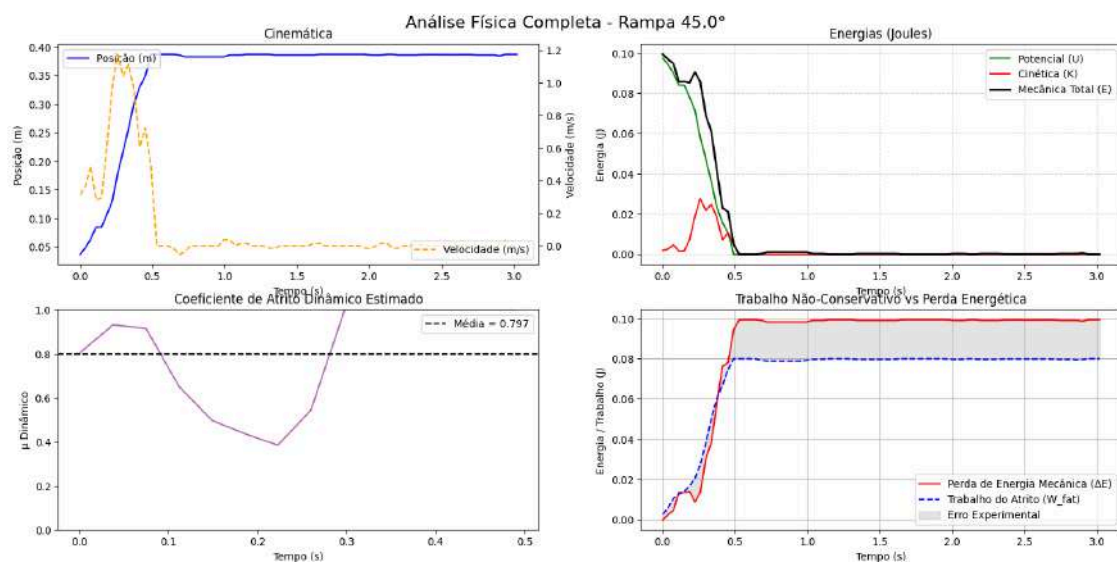


Figura 50 - 40g L80 CP80

O painel superior esquerdo de cada figura descreve a posição e a velocidade do corpo de prova em função do tempo a partir do início do movimento, o painel superior direito descreve a quantidade de energia teórica do corpo de prova em função do tempo. O painel inferior direito descreve a diferença entre a perda de energia teórica e a perda de energia experimental, por fim, o painel inferior esquerdo descreve o atrito dinâmico em função do tempo e a média calculada.

Pode-se notar certa variação nos erros experimentais das energias, facilmente explicada pela qualidade do sensor e motor utilizado. No entanto, é possível observar que o coeficiente de atrito dinâmico médio se mantém consistente nas relações de granulometria em que o atrito esperado seja igual, o que traz resultados satisfatórios.

## **6. Conclusão:**

Diante das limitações impostas em relação aos materiais serem de menor custo possível para equipe e que apresentasse resultados expressivos, o projeto realizado pela equipe foi concluído de maneira relevante, pois os materiais em sua maioria para construção do tribômetro foram aproveitados de trabalhos anteriores da disciplina de Mecânica dos Sólidos II, com o professor Roberto Bezerra, e de equipamentos eletrônicos já adquirido pelos integrantes do trabalho. Os resultados de rugosidade mostram com boa exatidão que o equipamento foi bem desenvolvido, apesar de suas limitações.

Para projetos futuros, a adição de um motor de passo que suporte um torque maior na ponta do eixo seria de grande relevância, com a sua adição teríamos mais segurança e colocar corpos de prova mais pesados sem temer de o motor não conseguir manter o ângulo travado, e com isso apresentaríamos uma automação mais robusta, como também a impressão em 3D de todas as partes que foram usadas em MDF, para entregar um melhor acabamento.

**Referências**

**BHUSHAN, B.** *Introduction to Tribology*. 2. ed. New York: John Wiley & Sons, 2013.

**BRASILESCOLA.** *Plano inclinado com atrito*. Disponível em:

<https://brasilescola.uol.com.br/fisica/plano-inclinado-com-atrito.htm>.

**BOLTON, W.** *Mecatrônica: uma abordagem multidisciplinar*. 4. ed. Porto Alegre: Bookman, 2010.

**BOWDEN, F. P.; TABOR, D.** *The Friction and Lubrication of Solids*. Oxford: Clarendon Press, 1950.

**HALLIDAY, D.; RESNICK, R.; WALKER, J.** *Fundamentos de física, volume 1: mecânica*. 9. ed. Rio de Janeiro: LTC, 2012.

**LUDEMA, K. C.** *Friction, Wear, Lubrication: A Textbook in Tribology*. CRC Press, 1996.

**MCROBERTS, M.** *Arduino básico*. São Paulo: Novatec, 2011.

**NORTON, R. L.** *Projeto de máquinas: uma abordagem integrada*. 4. ed. Porto Alegre: Bookman, 2013.

**RESEARCHGATE.** *Diagram illustrating the interference state of asperities between friction surfaces*.

Disponível em:

[https://www.researchgate.net/figure/Diagram-illustrating-the-interference-state-of-asperities-between-friction-surfaces\\_fig4\\_223224564](https://www.researchgate.net/figure/Diagram-illustrating-the-interference-state-of-asperities-between-friction-surfaces_fig4_223224564).

**USINAINFO.** *Balança Arduino com célula de peso e HX711: Tutorial calibrando e verificando peso*.

Disponível em:

<https://www.usinainfo.com.br/blog/balanca-arduino-com-celula-de-peso-e-hx711-tutorial-calibrando-e-verificando-peso/>.