Berechnen

February 17, 2024

1 Jupyter Notebook

1.1 Servo

1.1.1 Technische Spezifikationen

- **Motortyp**: Kernmotor.
- **Betriebsspannung**: 4,8 bis 6V DC.
- **Stromverbrauch**: <=50mA bei 4,8V, <=60mA bei 6V im Leerlauf.
- **Blockierstrom**: <=550mA bei 4,8V, <=650mA bei 6V.
- **Drehmoment**: >=0,6 kgf x cm bei 4,8V, >=0,7 kgf x cm bei 6V (Nennwert); >=1,4 kgf x cm bei 4,8V, >=1,6 kgf x cm bei 6V (Maximalwert).
- **Leerlaufdrehzahl**: <=0,14s/60° bei 4,8V; <=0,12s/60° bei 6V.
- **Betriebs- und Lagerungstemperatur**: -10 bis +50°C bzw. -20°C bis +60°C.
- **Feuchtigkeitsbereiche**: <= 90%RH für Betrieb und Lagerung.
- **Gewicht**: 10+/- 0,5g.
- **Material**: ABS.
- **Betriebswinkel**: 180°+/-10°.
- **Mechanischer Begrenzungswinkel**: 360°.

Drehmomentwerte des Servos in Newtonzentimeter (Ncm)

[ ]:
```
# Umrechnungsfaktor von kgf·cm in Ncm
umrechnungsfaktor_kgf_cm_zu_n_cm = 9.81

# Gegebene Drehmomentwerte in kgf·cm
drehmoment_nenn_4_8v_kgf_cm = 0.6
drehmoment_nenn_6v_kgf_cm = 0.7
drehmoment_max_4_8v_kgf_cm = 1.4
drehmoment_max_6v_kgf_cm = 1.6

# Umrechnung in Ncm
drehmoment_nenn_4_8v_n_cm = drehmoment_nenn_4_8v_kgf_cm *␣
 ↪umrechnungsfaktor_kgf_cm_zu_n_cm
drehmoment_nenn_6v_n_cm = drehmoment_nenn_6v_kgf_cm *␣
 ↪umrechnungsfaktor_kgf_cm_zu_n_cm
drehmoment_max_4_8v_n_cm = drehmoment_max_4_8v_kgf_cm *␣
 ↪umrechnungsfaktor_kgf_cm_zu_n_cm
```

```
drehmoment_max_6v_n_cm = drehmoment_max_6v_kgf_cm *␣
 ↪umrechnungsfaktor_kgf_cm_zu_n_cm

# Zusammenfassung der Berechnungen für 4.8V und 6V

# Ausgabe der Ergebnisse für 4.8V und 6V
ergebnisse = f"""
Zusammenfassung der Drehmomente:

4.8V:
- Nenndrehmoment: {drehmoment_nenn_4_8v_n_cm:.2f} N·cm
- Maximales Drehmoment: {drehmoment_max_4_8v_n_cm:.2f} N·cm

6V:
- Nenndrehmoment: {drehmoment_nenn_6v_n_cm:.2f} N·cm
- Maximales Drehmoment: {drehmoment_max_6v_n_cm:.2f} N·cm
"""

print(ergebnisse)
```

```
Zusammenfassung der Drehmomente:

4.8V:
- Nenndrehmoment: 5.89 N·cm
- Maximales Drehmoment: 13.73 N·cm

6V:
- Nenndrehmoment: 6.87 N·cm
- Maximales Drehmoment: 15.70 N·cm
```

1.2 RGB-LED-Streifen

1.2.1 Technische Merkmale

- Arbeitsspannung: DC5V.
- Vollfarbe RGB.
- Arbeitstemperatur: -15 °C bis 50 °C.
- RGB-Typ: 5050RGB.
- Strom: 150mA pro Einzelschaltkreis.
- Leistung: 1.5W.
- Dicke des Lichtstreifens: 2mm; Breite: 5.5mm.
- Anschlusskabel: ZH1.5-4P, 25cm Länge, 28AWG, in Schwarz.

[ ]:
```
# Definierte Konstanten
anzahl_pakete = 4
strom_pro_led = 150  # in mA
```

```
spannung = 5  # in V

# Gesamtstromverbrauch berechnen
gesamtstrom = anzahl_pakete * strom_pro_led  # in mA

# Gesamtleistung für ein Paket berechnen
leistung_einzel = spannung * strom_pro_led / 1000  # Umwandlung von mA in A␣
↪erforderlich

# Gesamtleistung für alle Pakete berechnen
gesamtleistung = anzahl_pakete * leistung_einzel  # in W

# Ergebnisse ausgeben
print(f"Gesamtstromverbrauch für {anzahl_pakete} Pakete: {gesamtstrom} mA")
print(f"Gesamtleistung für {anzahl_pakete} Pakete: {gesamtleistung} W")
```

```
Gesamtstromverbrauch für 4 Pakete: 600 mA
Gesamtleistung für 4 Pakete: 3.0 W
```

1.3 TT-Getriebemotor

1.3.1 Spezifikationen des TT-Getriebemotors

- **Spannungsbereich**: 3V bis 4.5V DC.
- **Anzahl der Wellen**: Einzelwelle.
- **Übersetzungsverhältnis**: 1:120.
- **Leerlaufstrom**: 130mA.
- **Leerlaufgeschwindigkeit**: 38rpm+/-8%rpm.
- **Startspannung**: Mindestens 2V im Leerlauf.
- **Ausgangsdrehmoment**: Bei 3V über 1.2kgf x cm.
- **Lebensdauer**: Zwischen 70 und 120 Stunden.
- **Drehrichtung**: Kann in beide Richtungen drehen.
- **Körperabmessungen**: 70 x 22,5 x 36,6mm.
- **Kabel**: Grau und Schwarz, 24AWG, 250mm Länge.
- **Stecker**: Weiß, XH2.54-2P.
- **Gewicht**: 28,5g.

[ ]:
```
# Umrechnung von kgf·cm in Ncm für ein Drehmoment von 1.2 kgf·cm
drehmoment_pro_motor_kgf_cm = 1.2
umrechnungsfaktor_kgf_cm_zu_n_cm = 9.81  # 1 kgf·cm entspricht 9.81 Ncm

# Berechnung des Drehmoments pro Motor in Ncm
drehmoment_pro_motor_n_cm = drehmoment_pro_motor_kgf_cm *␣
↪umrechnungsfaktor_kgf_cm_zu_n_cm

# Gesamtdrehmoment für sechs Motoren in Ncm
gesamtdrehmoment_n_cm = drehmoment_pro_motor_n_cm * 6
```

```
gesamtdrehmoment_n_cm
# Ergebnisse ausgeben
print(f"Gesamtdrehmoment von den sechs TT-DC-Getriebemotoren␣
↪{gesamtdrehmoment_n_cm} Ncm")
```

```
Gesamtdrehmoment von den sechs TT-DC-Getriebemotoren 70.632 Ncm
```

1.4 Solarpanel

- **Merkmale eines Solarpanels**:
  - **Ausgangsleistung**: 6V/660mA, was die Energiemenge angibt, die das Panel unter optimalen Bedingungen erzeugen kann.
  - **Ladezeit**: Es benötigt theoretisch 7,2 Stunden unter starkem Sonnenlicht, um einen Akku vollständig aufzuladen. Diese Zeit kann als Richtwert für die Effizienz des Panels dienen.
  - **Größe**: Mit Abmessungen von 170mm x 170mm ist es kompakt und vielseitig einsetzbar.
  - **Kabel und Stecker**: Ausgestattet mit einem grauen und schwarzen Kabel (200mm lang, 24AWG) und einem weißen XH2.54-2P Stecker, was die Installation und Verbindung erleichtert.

[ ]:
```
# Definieren der Ausgangsbedingungen
optimale_leistung_mA = 660  # in mA
ladezeit_optimal_h = 7.2  # in Stunden

# Berechnung für schwaches Sonnenlicht
leistung_schwach_mA = optimale_leistung_mA * 0.5  # 50% der optimalen Leistung
ladezeit_schwach_h = ladezeit_optimal_h * 2  # Doppelte Ladezeit

# Berechnung für bewölkte Bedingungen
leistung_bewoelkt_mA = optimale_leistung_mA * 0.25  # 25% der optimalen Leistung
ladezeit_bewoelkt_h = ladezeit_optimal_h * 4  # Vervierfachte Ladezeit

# Berechnung für Nacht
leistung_nacht_mA = 0  # Keine Leistung
ladezeit_nacht_h = "unendlich"  # Keine Aufladung möglich

# Ausgabe der Ergebnisse
rechenbeispiel = f"""
Rechenbeispiel unter verschiedenen Lichtverhältnissen:

Starkes Sonnenlicht:
- Ausgangsleistung: 6V/660mA
- Ladezeit: {ladezeit_optimal_h} Stunden

Schwaches Sonnenlicht (50%):
- Ausgangsleistung: 6V/{leistung_schwach_mA}mA
- Ladezeit: {ladezeit_schwach_h} Stunden

```

```
Bewölkt (25%):
- Ausgangsleistung: 6V/{leistung_bewoelkt_mA}mA
- Ladezeit: {ladezeit_bewoelkt_h} Stunden

Nacht:
- Ausgangsleistung: {leistung_nacht_mA}mA
- Ladezeit: {ladezeit_nacht_h}
"""

print(rechenbeispiel)
```

```
Rechenbeispiel unter verschiedenen Lichtverhältnissen:

Starkes Sonnenlicht:
- Ausgangsleistung: 6V/660mA
- Ladezeit: 7.2 Stunden

Schwaches Sonnenlicht (50%):
- Ausgangsleistung: 6V/330.0mA
- Ladezeit: 14.4 Stunden

Bewölkt (25%):
- Ausgangsleistung: 6V/165.0mA
- Ladezeit: 28.8 Stunden

Nacht:
- Ausgangsleistung: 0mA
- Ladezeit: unendlich
```

1.5 18650 Batterie

- **Aufbau**: Das Batteriepaket besteht aus zwei 18650 Zellen mit jeweils einer Kapazität von 2000mAh, was eine Gesamtkapazität von 4000mAh ergibt.
- **Anschlüsse**:
  - **VCC**: Positive Anschlüsse, vorhanden in zwei Sätzen, um den Stromfluss zu erhöhen und den Widerstand zu minimieren.
  - **Middle**: Dient der Spannungsausgleichung zwischen den Zellen, um die Batterie zu schützen.
  - **GND**: Negative Anschlüsse für den Stromkreis.
- **Anschlussart**: Verwendet wird ein XH2.54 3P-Anschluss, der eine direkte Aufladung des Batteriepakets ermöglicht, sobald es in ein kompatibles Shield eingesteckt wird.
- **Leistung**:
  - **Ladung**: Das Batteriepaket kann mit 5V/2A aufgeladen werden.
  - **Ausgang**: Es liefert einen Ausgang von 5V/5A, geeignet für Anwendungen, die eine hohe Stromstärke benötigen.
- **Spezifikationen**:

- **Batteriekapazität**: Jede Zelle hat eine Kapazität von 3.7V und 2000mAh, insgesamt also 4000mAh bei Zusammenschaltung.
- **Batterielebensdauer**: Das Paket ermöglicht eine Laufzeit von etwa 90 Minuten bei voller Ladung.
- **Ladezeit**: Die vollständige Aufladung des Batteriepakets dauert ca. 130 Minuten.

[ ]:
```
# Gegebene Daten
kapazitaet_mAh = 2000  # Kapazität einer der 18650 Batterien in mAh
ladestrom_mA = 2000  # Ladestrom in mA (5V/2A)

# Berechnung der Ladezeit in Stunden
ladezeit_stunden = kapazitaet_mAh / ladestrom_mA

# Umrechnung der Ladezeit in Minuten
ladezeit_minuten = ladezeit_stunden * 60

ladezeit_minuten
```

[ ]:
```
60.0
```

1.6 Batterielebensdauer und Entladestromstärke berechnen

Wenn die Batterie 90 Minuten lang hält, können wir den durchschnittlichen Entladestrom (I) berechnen, der für diese Laufzeit benötigt wird, mit der Formel:

[ ]:
```
# Gegebene Werte
kapazitaet_mAh = 2000  # Kapazität der Batterie in mAh
betriebsdauer_minuten = 90  # Betriebsdauer in Minuten

# Umrechnung der Betriebsdauer in Stunden
betriebsdauer_stunden = betriebsdauer_minuten / 60

# Berechnung des durchschnittlichen Entladestroms
entladestrom_mA = kapazitaet_mAh / betriebsdauer_stunden

# Ausgabe des Ergebnisses
print(f"Der durchschnittliche Entladestrom beträgt: {entladestrom_mA} mA, oder␣
↪{entladestrom_mA/1000:.2f} A.")
```

```
Der durchschnittliche Entladestrom beträgt: 1333.3333333333333 mA, oder 1.33 A.
```

1.7 Ultraschall - Berechne die Entfernung zu einem Objekt

Angenommen, wir messen die Entfernung zu einem Objekt, das 1 Meter entfernt ist.

1. **Trigger-Signal senden**:
   - Der Sensor erhält über den TRIG-Pin ein 10 Mikrosekunden (µs) hohes Signal, was ihn veranlasst, einen 8-Zyklus-Burst von Ultraschallwellen mit einer Frequenz von 40 kHz zu senden.

2. **Echo-Signal empfangen**:
   - Nach dem Senden des Bursts wartet der Sensor auf das Echo, also das Zurückkommen der Ultraschallwellen, nachdem diese vom Objekt reflektiert wurden.
3. **Zeitmessung**:
   - Die Zeit vom Senden bis zum Empfangen des Echos wird gemessen. Nehmen wir an, die gemessene Zeit (Echo-Zeit) beträgt 5.82 Millisekunden (ms).
4. **Entfernung berechnen**:

[ ]:
```python
# Gegebene Werte
echo_zeit_ms = 5.82  # Echo-Zeit in Millisekunden
schallgeschwindigkeit_m_s = 340  # Schallgeschwindigkeit in Luft in m/s

# Umrechnung der Echo-Zeit in Sekunden
echo_zeit_s = echo_zeit_ms * 0.001

# Berechnung der Entfernung
entfernung_m = (echo_zeit_s * schallgeschwindigkeit_m_s) / 2

# Ausgabe der Entfernung
print(f"Die Entfernung zum Objekt beträgt: {entfernung_m:.4f} Meter.")
```

```
Die Entfernung zum Objekt beträgt: 0.9894 Meter.
```

1.8 Mathematischen Zugang Ultraschall

Frequenz (f)

- **Formel**: $f = \frac{1}{T}$, wobei T die Periodendauer ist (die Zeit für eine vollständige Schwingung/Zyklus).

Hörbarkeit

- Der hörbare Bereich für Menschen liegt zwischen 20 Hz und 20.000 Hz. Frequenzen innerhalb dieses Bereichs sind hörbar, Frequenzen außerhalb dieses Bereichs sind entweder Ultraschall ($>$ 20 kHz) oder Infraschall ($<$ 20 Hz).

Schallwellen

- **Geschwindigkeit (v) einer Welle**: $v = f\lambda$, wobei λ die Wellenlänge ist.
- **Schallgeschwindigkeit in Luft**: etwa 343 m/s bei 20 °C.

Hertz (Hz)

- Einheit für Frequenz, 1 Hz = 1 Schwingung pro Sekunde.

Hörschwelle

- Die Hörschwelle variiert je nach Frequenz, wobei etwa 0 dB SPL (Schalldruckpegel) für Frequenzen um 1 kHz als leiseste hörbare Töne für Menschen gelten.

Ultraschall und Infraschall

- **Ultraschall**: Frequenzen > 20 kHz, nicht hörbar für Menschen.
- **Infraschall**: Frequenzen < 20 Hz, ebenfalls nicht hörbar für Menschen.

Dezibel (dB)

- **Formel zur Berechnung des Schalldruckpegels**: $L = 20\log_{10}(\frac{p}{p_0})$, wobei p der Schalldruck und p_0 der Referenzschalldruck (üblicherweise 20 µPa in Luft) ist.

Schallgeschwindigkeit

- Abhängig vom Medium. In Luft etwa 343 m/s, in Wasser etwa 1.483 m/s und in Stahl etwa 5.960 m/s.

Echoortung und Sonar

- **Entfernungsbestimmung**: $D = \frac{1}{2}tv$, wobei t die Zeit zwischen Aussenden und Empfangen des Echos ist und v die Schallgeschwindigkeit im Medium.

Frequenzmodulation und -demodulation

- **Frequenzmodulation (FM)**: $f(t) = f_c + \Delta f \sin(2\pi f_m t)$, wobei f_c die Trägerfrequenz, Δf die Frequenzabweichung und f_m die Modulationsfrequenz ist.

Dopplereffekt

- **Formel für bewegten Beobachter**: $f' = f\left(\frac{v \pm v_o}{v \mp v_s}\right)$, wobei f' die wahrgenommene Frequenz, f die ursprüngliche Frequenz, v die Schallgeschwindigkeit, v_o die Geschwindigkeit des Beobachters und v_s die Geschwindigkeit der Quelle ist. Pluszeichen wird verwendet, wenn sich Beobachter und Quelle einander nähern, Minuszeichen, wenn sie sich voneinander entfernen.

[ ]:
```python
from math import log10, sin, pi

# Frequenzberechnung
def berechne_frequenz(T):
    return 1 / T

# Schallwellengeschwindigkeit
def geschwindigkeit_schallwelle(f, lambda_welle):
    return f * lambda_welle

# Schalldruckpegel in Dezibel
def berechne_dezibel(p, p0=20e-6):
    return 20 * log10(p / p0)

# Echoortung und Sonar zur Entfernungsbestimmung
def berechne_entfernung(t, v=343):
    return 0.5 * t * v
```

```python
# Frequenzmodulation
def frequenzmodulation(f_c, Delta_f, f_m, t):
    return f_c + Delta_f * sin(2 * pi * f_m * t)

# Dopplereffekt
def dopplereffekt(f, v=343, v_o=0, v_s=0, annaehrung=True):
    if annaehrung:
        return f * ((v + v_o) / (v - v_s))
    else:
        return f * ((v - v_o) / (v + v_s))

# Beispielwerte
T = 0.000025  # Periodendauer in Sekunden (40 kHz Ultraschallwelle)
lambda_welle = 0.0085  # Wellenlänge in Metern
p = 0.002  # Schalldruck in Pascal
t = 0.01  # Zeit in Sekunden für Echoortung
f_c = 1000  # Trägerfrequenz in Hz
Delta_f = 50  # Frequenzabweichung in Hz
f_m = 100  # Modulationsfrequenz in Hz
v_beobachter = 5  # Geschwindigkeit des Beobachters in m/s
v_quelle = 0  # Geschwindigkeit der Quelle in m/s

# Berechnungen durchführen
frequenz = berechne_frequenz(T)
geschwindigkeit = geschwindigkeit_schallwelle(frequenz, lambda_welle)
dezibel = berechne_dezibel(p)
entfernung = berechne_entfernung(t)
fm_signal = frequenzmodulation(f_c, Delta_f, f_m, t)
doppler_annaehrung = dopplereffekt(f_c, v_o=v_beobachter, v_s=v_quelle, annaehrung=True)
doppler_entfernung = dopplereffekt(f_c, v_o=v_beobachter, v_s=v_quelle, annaehrung=False)

# Ergebnisse ausgeben
print(f"Frequenz: {frequenz} Hz")
print(f"Geschwindigkeit der Schallwelle: {geschwindigkeit} m/s")
print(f"Schalldruckpegel: {dezibel} dB")
print(f"Entfernung durch Echoortung: {entfernung} m")
print(f"Frequenzmoduliertes Signal: {fm_signal} Hz")
print(f"Dopplereffekt bei Annäherung: {doppler_annaehrung} Hz")
print(f"Dopplereffekt bei Entfernung: {doppler_entfernung} Hz")
```

```
Frequenz: 40000.0 Hz
Geschwindigkeit der Schallwelle: 340.0 m/s
Schalldruckpegel: 40.0 dB
Entfernung durch Echoortung: 1.715 m
Frequenzmoduliertes Signal: 1000.0 Hz
```

```
Dopplereffekt bei Annäherung: 1014.5772594752187 Hz
Dopplereffekt bei Entfernung: 985.4227405247813 Hz
```

1.9 Mathematischen Zugang Infrarotlicht (IR)

1. Wellenlänge (λ)

- **Mathematische Darstellung**: $\lambda = \frac{v}{f}$
  - v ist die Geschwindigkeit der Welle (z.B. Lichtgeschwindigkeit im Vakuum, etwa 3×10^8 m/s).
  - f ist die Frequenz der Welle in Hertz (Hz).

2. Sichtbares Licht und Elektromagnetisches Spektrum

- **Bereich des sichtbaren Lichts**: Etwa 380 nm bis 740 nm.
- **Umrechnung in Frequenz**: $f = \frac{c}{\lambda}$, wobei c die Lichtgeschwindigkeit ist.

3. Infrarotlicht (IR) und Ultraviolettes Licht (UV)

- **IR-Bereich**: Wellenlängen größer als 740 nm.
- **UV-Bereich**: Wellenlängen kleiner als 380 nm.
- **Berechnung der Energie eines Photons**: $E = hf = \frac{hc}{\lambda}$, wobei h das Plancksche Wirkungsquantum ist.

4. Nanometer (nm)

- **Umrechnung**: 1 nm = 10^{-9} Meter.
- **Anwendung in Formeln**: Direkte Einsetzung der Wellenlänge in nm nach Umrechnung in Meter für physikalische Berechnungen.

5. Transparenz, Reflexion, Brechung, und Lichtabsorption

- **Brechungsindex**: $n = \frac{c}{v_{medium}}$, wobei v_{medium} die Geschwindigkeit des Lichts im Medium ist.
- **Snelliussches Brechungsgesetz**: $n_1 \sin(\theta_1) = n_2 \sin(\theta_2)$, wobei θ der Winkel zur Normalen ist.

6. Prisma und Lichtdispersion

- **Dispersion**: Abhängigkeit des Brechungsindex von der Wellenlänge, was zu Farbaufspaltung führt.

7. Farbtemperatur

- **Einheit**: Kelvin (K).
- **Zusammenhang mit Spektrum**: Höhere Farbtemperaturen entsprechen bläulicherem Licht, niedrigere wärmerem Licht.

8. **Lasertechnologie und Holographie**

- **Kohärenz und Monochromasie**: Laserlicht hat sehr enge Wellenlängenbereiche, ideal für Interferenz und Holographie.

9. **Photonik und Elektromagnetische Wellentheorie**

- **Photonenenergie**: Siehe Formel oben.
- **Maxwellsche Gleichungen**: Grundlage der elektromagnetischen Wellentheorie, beschreiben die Ausbreitung elektromagnetischer Felder.

[ ]:
```python
# Konstanten
lichtgeschwindigkeit_c = 3e8  # Lichtgeschwindigkeit im Vakuum in m/s
planck_sches_wirkungsquantum_h = 6.62607015e-34  # Plancksches Wirkungsquantum␣
↪in Joule·Sekunden

# Wellenlänge aus Frequenz berechnen
def wellenlaenge_aus_frequenz(f):
    return lichtgeschwindigkeit_c / f

# Frequenz aus Wellenlänge berechnen
def frequenz_aus_wellenlaenge(lambda_nm):
    lambda_m = lambda_nm * 1e-9  # Umrechnung von nm in m
    return lichtgeschwindigkeit_c / lambda_m

# Energie eines Photons berechnen
def energie_photon(lambda_nm):
    lambda_m = lambda_nm * 1e-9  # Umrechnung von nm in m
    return (plancksches_wirkungsquantum_h * lichtgeschwindigkeit_c) / lambda_m

# Beispiele
lambda_ir = 800  # Wellenlänge von IR-Licht in nm
lambda_uv = 300  # Wellenlänge von UV-Licht in nm

# Berechnungen
frequenz_ir = frequenz_aus_wellenlaenge(lambda_ir)
frequenz_uv = frequenz_aus_wellenlaenge(lambda_uv)
energie_photon_ir = energie_photon(lambda_ir)
energie_photon_uv = energie_photon(lambda_uv)

# Ergebnisse ausgeben
print(f"Frequenz von IR-Licht (800 nm): {frequenz_ir:.2e} Hz")
print(f"Frequenz von UV-Licht (300 nm): {frequenz_uv:.2e} Hz")
print(f"Energie eines Photons im IR-Bereich (800 nm): {energie_photon_ir:.2e}␣
↪J")
print(f"Energie eines Photons im UV-Bereich (300 nm): {energie_photon_uv:.2e}␣
↪J")
```

```
Frequenz von IR-Licht (800 nm): 3.75e+14 Hz
```

```
Frequenz von UV-Licht (300 nm): 1.00e+15 Hz
Energie eines Photons im IR-Bereich (800 nm): 2.48e-19 J
Energie eines Photons im UV-Bereich (300 nm): 6.63e-19 J
```

1.10 Newton

Ein **Newton** (Symbol: **N**) ist die SI-Einheit der Kraft. Sie ist definiert als die Kraft, die erforderlich ist, um einem Kilogramm Masse eine Beschleunigung von einem Meter pro Sekunde im Quadrat zu erteilen. Mathematisch ausgedrückt:

$1\,\text{N} = 1\,\text{kg} \cdot \text{m/s}^2$

Das bedeutet, dass ein Newton die Menge an Kraft ist, die benötigt wird, um ein Objekt mit einer Masse von einem Kilogramm mit einer Rate von einem Meter pro Sekunde pro Sekunde zu beschleunigen.

[ ]:
```python
# Gegebene Werte
m = 1000  # Masse in Kilogramm
delta_v = 10  # Geschwindigkeitsänderung in m/s
delta_t = 5  # Zeitdauer in Sekunden

# Beschleunigung berechnen
a = delta_v / delta_t

# Kraft berechnen
F = m * a

# Ausgabe der Ergebnisse
print(f"Beschleunigung: {a} m/s^2")
print(f"Erforderliche Kraft: {F} N")
```

```
Beschleunigung: 2.0 m/s^2
Erforderliche Kraft: 2000.0 N
```

1.11 Mathematischen Zugang Gravitation, Newtonsche Gesetze und die grundlegenden physikalischen Einheiten

Gravitation

- **Newtonsches Gravitationsgesetz**: $F = G\frac{m_1 m_2}{r^2}$
  - **Beispiel**: Zwei Massen von je 1 kg im Abstand von 1 m zueinander üben eine Anziehungskraft von etwa $6,674 \times 10^{-11}$ N aus.

Newton

- **Zweites Newtonsches Gesetz**: $F = m \cdot a$
  - **Beispiel**: Ein Objekt mit einer Masse von 10 kg, das mit $2\,\text{m/s}^2$ beschleunigt wird, erfährt eine Kraft von $20\,\text{N}$.

Kraft

- **Kraftvektor**: $F = m \cdot a$
  - **Beispiel**: Ein Auto (Masse = 1000 kg) beschleunigt mit $1\,\mathrm{m/s}^2$, die wirkende Kraft ist $1000\,\mathrm{N}$.

Beschleunigung

- **Gleichmäßige Beschleunigung**: $a = \frac{\Delta v}{\Delta t}$
  - **Beispiel**: Ein Auto erhöht seine Geschwindigkeit von 0 auf $60\,\mathrm{km/h}$ (etwa $16,67\,\mathrm{m/s}$) in 5 Sekunden, $a = 3,33\,\mathrm{m/s}^2$.

Masse und Gewicht

- **Gewicht**: $W = m \cdot g$
  - **Beispiel**: Eine Masse von 50 kg auf der Erde (mit $g = 9,81\,\mathrm{m/s}^2$) hat ein Gewicht von $490,5\,\mathrm{N}$.

Meter und Kilogramm

- **Längenmessung**: 1 Meter ist die Basislänge.
- **Masse**: 1 Kilogramm ist die Basismasse.

Sekunde, Ampere, Kelvin, Mol, Candela

- **Zeit**: 1 Sekunde ist die Basiseinheit der Zeit.
- **Stromstärke**: 1 Ampere ist die Basiseinheit der elektrischen Stromstärke.
- **Temperatur**: 1 Kelvin ist die Basiseinheit der Temperatur.
- **Stoffmenge**: 1 Mol ist die Basiseinheit der Stoffmenge.
- **Lichtstärke**: 1 Candela ist die Basiseinheit der Lichtstärke.

Fallgesetze und Impulserhaltung

- **Freier Fall**: $s = \frac{1}{2}gt^2$
  - **Beispiel**: Ein Objekt fällt für 2 Sekunden, zurückgelegte Strecke ist $19,62\,\mathrm{m}$.

Arbeit und Energie

- **Arbeit**: $W = F \cdot d$
  - **Beispiel**: Heben eines Objekts (Kraft = 100 N) um 2 Meter verrichtet eine Arbeit von $200\,\mathrm{J}$.

Drehmoment

- **Drehmoment**: $\tau = F \cdot r$
  - **Beispiel**: Eine Kraft von $10\,\mathrm{N}$, die im Abstand von 0,5 m vom Drehpunkt wirkt, erzeugt ein Drehmoment von $5\,\mathrm{N} \cdot \mathrm{m}$.

Schwingungen und Wellen

- **Frequenz einer Welle**: $f = \frac{1}{T}$
  - **Beispiel**: Eine Schwingung, die sich alle 0,5 Sekunden wiederholt, hat eine Frequenz von $2\,\mathrm{Hz}$.

[ ]:

```python
G = 6.674 * 10**-11  # Gravitationskonstante in N(m^2)/(kg^2)
m_1 = 1  # Masse in kg
m_2 = 1  # Masse in kg
r = 1  # Abstand in m

# Newtonsches Gravitationsgesetz
F_gravitation = G * (m_1 * m_2) / r**2

# Zweites Newtonsches Gesetz
m = 10  # Masse in kg
a = 2  # Beschleunigung in m/s^2
F_newton = m * a

# Kraftvektor
m_auto = 1000  # Masse des Autos in kg
a_auto = 1  # Beschleunigung des Autos in m/s^2
F_auto = m_auto * a_auto

# Beschleunigung
delta_v = 16.67  # Geschwindigkeitsänderung in m/s (60 km/h in m/s umgerechnet)
delta_t = 5  # Zeitdauer in s
a_auto_beschleunigung = delta_v / delta_t

# Masse und Gewicht
m_gewicht = 50  # Masse in kg
g = 9.81  # Erdbeschleunigung in m/s^2
W = m_gewicht * g

# Freier Fall
t_fall = 2  # Zeit in Sekunden
s_fall = 0.5 * g * t_fall**2

# Arbeit
F_heben = 100  # Kraft in N
d_heben = 2  # Distanz in m
W_heben = F_heben * d_heben

# Drehmoment
F_dreh = 10  # Kraft in N
r_dreh = 0.5  # Abstand zum Drehpunkt in m
tau = F_dreh * r_dreh

# Frequenz einer Welle
T_welle = 0.5  # Periodendauer in s
f_welle = 1 / T_welle

# Ergebnisse ausgeben
```

```python
print(f"Gravitationskraft zwischen zwei 1kg Massen: {F_gravitation} N")
print(f"Kraft auf ein 10kg Objekt mit 2 m/s^2 Beschleunigung: {F_newton} N")
print(f"Kraft, um ein Auto mit 1000 kg zu beschleunigen: {F_auto} N")
print(f"Beschleunigung eines Autos: {a_auto_beschleunigung} m/s^2")
print(f"Gewicht einer 50kg Masse auf der Erde: {W} N")
print(f"Zurückgelegte Strecke im freien Fall nach 2 Sekunden: {s_fall} m")
print(f"Arbeit, um ein Objekt um 2 Meter zu heben: {W_heben} J")
print(f"Drehmoment durch eine 10N Kraft: {tau} Nm")
print(f"Frequenz einer Schwingung mit 0,5s Periodendauer: {f_welle} Hz")
```

```
Gravitationskraft zwischen zwei 1kg Massen: 6.674e-11 N
Kraft auf ein 10kg Objekt mit 2 m/s^2 Beschleunigung: 20 N
Kraft, um ein Auto mit 1000 kg zu beschleunigen: 1000 N
Beschleunigung eines Autos: 3.3340000000000005 m/s^2
Gewicht einer 50kg Masse auf der Erde: 490.5 N
Zurückgelegte Strecke im freien Fall nach 2 Sekunden: 19.62 m
Arbeit, um ein Objekt um 2 Meter zu heben: 200 J
Drehmoment durch eine 10N Kraft: 5.0 Nm
Frequenz einer Schwingung mit 0,5s Periodendauer: 2.0 Hz
```

1.12 Gravitationsbeschleunigungen für Erde, Mond und Mars

Die Gravitationsbeschleunigung eines Himmelskörpers hängt hauptsächlich von zwei Faktoren ab:

1. **Masse des Himmelskörpers**: Je größer die Masse eines Himmelskörpers ist, desto stärker ist seine Gravitationskraft. Nach dem Gravitationsgesetz von Newton ist die Gravitationskraft direkt proportional zur Masse des Himmelskörpers. Das bedeutet, dass ein Planet mit einer höheren Masse eine stärkere Gravitationsanziehung hat, was zu einer höheren Gravitationsbeschleunigung führt.

2. **Radius des Himmelskörpers**: Die Gravitationsbeschleunigung hängt auch invers quadratisch vom Radius des Himmelskörpers ab. Wenn man sich weiter vom Zentrum eines Himmelskörpers entfernt, nimmt die Gravitationsbeschleunigung ab. Dies bedeutet, dass bei zwei Himmelskörpern mit gleicher Masse der Körper mit dem kleineren Radius eine höhere Gravitationsbeschleunigung an seiner Oberfläche haben wird, da man näher am Massenzentrum ist.

Mathematisch wird die Gravitationsbeschleunigung (g) an der Oberfläche eines Himmelskörpers durch die Formel

$g = \frac{G \cdot M}{R^2}$

ausgedrückt, wobei:

- G die universelle Gravitationskonstante ist ($6,674 \times 10^{-11}\,\mathrm{m}^3\mathrm{kg}^{-1}\mathrm{s}^{-2}$),
- M die Masse des Himmelskörpers ist,
- R der Radius des Himmelskörpers ist.

[ ]:
```python
# Gegebene Werte für Erde, Mond und Mars
# Universelle Gravitationskonstante
G = 6.674 * 10**-11  # m^3 kg^-1 s^-2
```

```python
# Erde
M_Erde = 5.972 * 10**24  # kg
R_Erde = 6.371 * 10**6  # m

# Mond
M_Mond = 7.347 * 10**22  # kg
R_Mond = 1.737 * 10**6  # m

# Mars
M_Mars = 6.417 * 10**23  # kg
R_Mars = 3.3895 * 10**6  # m

# Berechnung der Gravitationsbeschleunigung für Erde, Mond und Mars
g_Erde = (G * M_Erde) / (R_Erde**2)
g_Mond = (G * M_Mond) / (R_Mond**2)
g_Mars = (G * M_Mars) / (R_Mars**2)

# Ausgabe der Gravitationsbeschleunigungen für Erde, Mond und Mars
print("Gravitationsbeschleunigung auf der Erde:", g_Erde, "m/s^2")
print("Gravitationsbeschleunigung auf dem Mond:", g_Mond, "m/s^2")
print("Gravitationsbeschleunigung auf dem Mars:", g_Mars, "m/s^2")
```

```
Gravitationsbeschleunigung auf der Erde: 9.819532032815959 m/s^2
Gravitationsbeschleunigung auf dem Mond: 1.6251617990241847 m/s^2
Gravitationsbeschleunigung auf dem Mars: 3.7277515985747827 m/s^2
```

1.13 Mathematischer Zugang zu SI-Einheiten

Meter (m) - Einheit der Länge

- **Beispiel**: Ein Raum ist 5 m lang. Wenn man den Boden mit Fliesen auslegt, die 0,25 m breit sind, benötigt man $5\,\mathrm{m}/0,25\,\mathrm{m} = 20$ Fliesen für eine Reihe.

Kilogramm (kg) - Einheit der Masse

- **Beispiel**: Ein Sack Reis wiegt 25 kg. Wenn eine Person täglich 0,5 kg Reis verbraucht, reicht der Sack für $25\,\mathrm{kg}/0,5\,\mathrm{kg/Tag} = 50$ Tage.

Sekunde (s) - Einheit der Zeit

- **Beispiel**: Ein Lichtsignal benötigt 3 Sekunden, um von einem Sender zu einem Empfänger zu gelangen. Die Übertragungsgeschwindigkeit des Signals beträgt also $1/3\,\mathrm{s}^{-1}$.

Ampere (A) - Einheit der elektrischen Stromstärke

- **Beispiel**: Ein elektrisches Gerät hat eine Stromaufnahme von 2 A. Wenn das Gerät für 1 Stunde eingeschaltet ist, fließt ein elektrischer Strom von 2 Amperestunden (Ah) durch das Gerät.

Kelvin (K) - Einheit der thermodynamischen Temperatur

- **Beispiel**: Die Zimmertemperatur beträgt 293 K. Dies entspricht etwa 20 °C, da $20°C = 293K - 273,15$.

Mol (mol) - Einheit der Stoffmenge

- **Beispiel**: 1 mol Wasser (H O) enthält etwa $6,022 \times 10^{23}$ Moleküle und hat eine Masse von etwa 18 g.

Candela (cd) - Einheit der Lichtstärke

- **Beispiel**: Eine Kerze hat eine Lichtstärke von etwa 1 cd. Wenn man einen Raum mit 20 Kerzen beleuchtet, erreicht man eine Gesamtlichtstärke von 20 cd.

[ ]:
```python
# Berechnungen für gegebene Beispiele der SI-Einheiten

# Meter
laenge_raum_m = 5  # Länge des Raumes in Metern
breite_fliese_m = 0.25  # Breite der Fliese in Metern
anzahl_fliesen = laenge_raum_m / breite_fliese_m

# Kilogramm
masse_sack_kg = 25  # Masse des Sacks in Kilogramm
taeglicher_verbrauch_kg = 0.5  # Täglicher Verbrauch in kg
reichweite_tage = masse_sack_kg / taeglicher_verbrauch_kg

# Sekunde
zeit_s = 3  # Zeit in Sekunden
uebertragungsgeschwindigkeit_inv_s = 1 / zeit_s

# Ampere
stromaufnahme_A = 2  # Stromaufnahme in Ampere
dauer_h = 1  # Dauer in Stunden
strommenge_Ah = stromaufnahme_A * dauer_h

# Kelvin
zimmertemperatur_K = 293  # Zimmertemperatur in Kelvin
zimmertemperatur_C = zimmertemperatur_K - 273.15  # Umrechnung in Celsius

# Mol
anzahl_molekuele_pro_mol = 6.022 * 10**23  # Avogadro-Konstante
masse_wasser_g = 18  # Masse von 1 mol Wasser in Gramm

# Candela
lichtstaerke_kerze_cd = 1  # Lichtstärke einer Kerze in Candela
anzahl_kerzen = 20  # Anzahl der Kerzen
gesamtlichtstaerke_cd = lichtstaerke_kerze_cd * anzahl_kerzen
```

```python
# Ergebnisse ausgeben
print(f"Anzahl der Fliesen für eine Reihe: {anzahl_fliesen}")
print(f"Reichweite des Sacks Reis: {reichweite_tage} Tage")
print(f"Übertragungsgeschwindigkeit des Lichtsignals:␣
↪{uebertragungsgeschwindigkeit_inv_s} s^-1")
print(f"Elektrische Strommenge, die durch das Gerät fließt: {strommenge_Ah} Ah")
print(f"Zimmertemperatur in °C: {zimmertemperatur_C} °C")
print(f"Anzahl der Moleküle in 1 mol Wasser: {anzahl_molekuele_pro_mol}␣
↪Moleküle")
print(f"Gesamtlichtstärke der Kerzen: {gesamtlichtstaerke_cd} cd")
```

```
Anzahl der Fliesen für eine Reihe: 20.0
Reichweite des Sacks Reis: 50.0 Tage
Übertragungsgeschwindigkeit des Lichtsignals: 0.3333333333333333 s^-1
Elektrische Strommenge, die durch das Gerät fließt: 2 Ah
Zimmertemperatur in °C: 19.850000000000023 °C
Anzahl der Moleküle in 1 mol Wasser: 6.0219999999999996e+23 Moleküle
Gesamtlichtstärke der Kerzen: 20 cd
```

1.14 Was ist ein Joule?

Ein Joule (Symbol: J) ist die SI-Einheit der Energie, benannt nach dem englischen Physiker James Prescott Joule. Ein Joule ist definiert als die Energiemenge, die benötigt wird, um ein Objekt mit einer Kraft von einem Newton einen Meter weit zu bewegen. Es ist auch die Energiemenge, die verbraucht wird, wenn ein Strom von einem Ampere eine Sekunde lang durch einen Widerstand von einem Ohm fließt. In anderen Worten:

$1\,\mathrm{J} = 1\,\mathrm{N} \cdot \mathrm{m} = 1\,\mathrm{W} \cdot \mathrm{s} = 1\,\mathrm{kg} \cdot \mathrm{m}^2 \cdot \mathrm{s}^{-2}$

1.14.1 Kinetische Energie

Kinetische Energie ist die Energie, die ein Objekt aufgrund seiner Bewegung besitzt. Sie hängt von der Masse des Objekts und seiner Geschwindigkeit ab. Die Formel für die kinetische Energie (E_k) eines Objekts lautet:

$E_k = \frac{1}{2}mv^2$

wo:

- E_k die kinetische Energie in Joule (J) ist,
- m die Masse des Objekts in Kilogramm (kg),
- v die Geschwindigkeit des Objekts in Metern pro Sekunde (m/s).

[ ]:
```python
# Kinetische Energie eines Autos berechnen
# Anpassung des Programms zur Ausgabe der kinetischen Energie in sinnvollen␣
↪Einheitengrößen (Joule und Megajoule)

# Definition der Umrechnungsfunktion von km/h in m/s
def kmh_to_ms(velocity_kmh):
    return velocity_kmh / 3.6
```

```
# Definition der Funktion zur Berechnung der kinetischen Energie
def calculate_kinetic_energy(mass_kg, velocity_ms):
    return 0.5 * mass_kg * velocity_ms**2

# Umrechnung der Energie von Joule in Megajoule
def joule_to_megajoule(energie_j):
    return energie_j / 1_000_000

# Masse des Autos in Kilogramm
mass_kg = 1500  # Beispielhafte Masse eines Autos

# Verschiedene Geschwindigkeiten in km/h
geschwindigkeiten_kmh = [50, 80, 130, 200]

# Berechnung und Ausgabe der kinetischen Energie für jede Geschwindigkeit in␣
↪Joule und Megajoule
for geschwindigkeit in geschwindigkeiten_kmh:
    # Umrechnung der Geschwindigkeit in m/s
    geschwindigkeit_ms = kmh_to_ms(geschwindigkeit)
    # Berechnung der kinetischen Energie
    energie_j = calculate_kinetic_energy(mass_kg, geschwindigkeit_ms)
    energie_mj = joule_to_megajoule(energie_j)
    print(f"Kinetische Energie eines {mass_kg} kg schweren Autos bei␣
↪{geschwindigkeit} km/h: {energie_j:.2f} Joule ({energie_mj:.2f} MJ)")
```

```
Kinetische Energie eines 1500 kg schweren Autos bei 50 km/h: 144675.93 Joule
(0.14 MJ)
Kinetische Energie eines 1500 kg schweren Autos bei 80 km/h: 370370.37 Joule
(0.37 MJ)
Kinetische Energie eines 1500 kg schweren Autos bei 130 km/h: 978009.26 Joule
(0.98 MJ)
Kinetische Energie eines 1500 kg schweren Autos bei 200 km/h: 2314814.81 Joule
(2.31 MJ)
```

[ ]:
```
# Berechne Zunahme von Geschwindigkeit und kinetischer Energie mit der Zeit im␣
↪freien Fall
# Berechnungen für verschiedene Zeiten: 1s, 2s, 10s, 20s, 60s
zeiten_s = [1, 2, 10, 20, 60]
masse_kg = 80  # Masse in Kilogramm

# Listen für Ergebnisse
strecken_m = []
geschwindigkeiten_m_s = []
energien_j = []

for t in zeiten_s:
```

```python
    # Berechnung der zurückgelegten Strecke
    s = 0.5 * g * t**2
    strecken_m.append(s)

    # Berechnung der Geschwindigkeit am Ende der Zeitperiode
    v = g * t
    geschwindigkeiten_m_s.append(v)

    # Berechnung der kinetischen Energie am Ende der Zeitperiode
    E_kin = 0.5 * masse_kg * v**2
    energien_j.append(E_kin)

# Ausgabe der Ergebnisse
for t, s, v, E in zip(zeiten_s, strecken_m, geschwindigkeiten_m_s, energien_j):
    print(f"Zeit: {t} Sekunde(n), Strecke: {s:.2f} Meter, Geschwindigkeit: {v:.
 ↪2f} m/s, Kinetische Energie: {E:.2f} Joule")
```

```
Zeit: 1 Sekunde(n), Strecke: 4.91 Meter, Geschwindigkeit: 9.81 m/s, Kinetische
Energie: 3849.44 Joule
Zeit: 2 Sekunde(n), Strecke: 19.62 Meter, Geschwindigkeit: 19.62 m/s, Kinetische
Energie: 15397.78 Joule
Zeit: 10 Sekunde(n), Strecke: 490.50 Meter, Geschwindigkeit: 98.10 m/s,
Kinetische Energie: 384944.40 Joule
Zeit: 20 Sekunde(n), Strecke: 1962.00 Meter, Geschwindigkeit: 196.20 m/s,
Kinetische Energie: 1539777.60 Joule
Zeit: 60 Sekunde(n), Strecke: 17658.00 Meter, Geschwindigkeit: 588.60 m/s,
Kinetische Energie: 13857998.40 Joule
```