2022

Tugas Besar 1 MK Pemodelan dan Simulasi

ANALISIS GERAK PELURU DENGAN HAMBATAN UDARA

DIO ADITYA PUTRA WARDHONO

TELKOM UNIVERSITY | 1301194290

Daftar Gambar

Daftar Isi

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pergerakan suatu objek yang ditembakkan dari ketinggian tertentu dapat didekati dengan menggunakan konsep gerak peluru. Pada kajian yang sederhana, pergerakan objek pada gerak peluru dapat dihitung dengan mengabaikan hambatan udara. Pada kasus yang lebih riil, simulasi gerak peluru perlu mempertimbangkan faktor hambatan udara yang mempengaruhi pergerakan objek.[1] Oleh karena itu, saya akan melakukan beberapa penelitian terkait dengan gerak peluru dengan 2 metode yaitu, metode analitik dan metode numerik untuk kedua kasus tersebut jika suatu objek dengan massa 0.15 kg ditembakkan dari permukaan tanah dengan kecepatan awal 50 m/s dan sudut tembak 35° . Dengan menggunakan nilai D = 0.0013 dan Δt = 0.01, dimana g. adalah percepatan gravitasi yang nilainya -9.806 m/s^2.[1] Penelitian singkat ini juga dibuat untuk memenuhi tugas besar dari mata kuliah pemodelan dan simulasi universitas Telkom Falkultas Informatika Jurusan Informatika.

1.2 Rumusan Masalah

Beberapa masalah akan saya jabarkan terkait untuk memnuhi penelitian ini dan juga sebagai jawaban untuk soal-soal yang disyaratkan dalam spesifikasi tugas besar pemodelan dan simulasi. Permasalahan tersebut, antara lain:

- Berapa posisi dan perbedaan objek sejak ditembakkan hingga sampai ke permukaan tanah dengan mengabaikan dan mempertimbangkan hambatan udara?
- Bagaimana validasi terhadap hasil perhitungan numerik untuk kasus yang pertama (tanpa hambatan udara)?

1.3 Tujuan Masalah

Berdasarkan rumusan masalah yang sudah saya jabarkan maka dapat disimpulkan tujuan dari laporan penelitian ini, antara lain:

- Untuk mengetahui posisi objek sejak ditembakkan hingga sampai ke permukaan tanah dengan mengabaikan dan mempertimbangkan hambatan udara;
- Untuk mengetahui perbandingan antara posisi objek sejak ditembakkan hingga sampai ke permukaan tanah dengan mengabaikan dan mempertimbangkan hambatan udara;
- Untuk mengetahui bagaimana cara Bagaimana validasi terhadap hasil perhitungan numerik untuk kasus yang pertama (tanpa hambatan udara).

1.4 Metode Penyelesaian

Metode penyelesaian yang akan ditempuh penulis untuk menyelesaian laporan penelitian tugas besar pemodelan dan simulasi adalah dengan menggunakan metode pendekatan secara analitik dan numerik dimana di tiap-tiap pendekatan akan saya jabarkan rumus-rumus matematikanya kemudian baru saya akan menjelaskan konversi rumus-rumus tersebut menjadi algoritma pemograman. Setelah itu, algoritma tersebut akan diimplementasikan menggunakan bahasa pemograman *Python* yang kemudian akan disimulasikan atau digambarkan dengan library *Matplotlib*.

BAB II
PEMBAHASAN

2.1 Pendahuluan Gerak Peluru

- Kinematika
  Kinematika adalah ilmu menggambarkan gerak benda dengan menggunakan kata-kata, diagram, angka, grafik, dan persamaan. Kinematika adalah cabang dari mekanika. Tujuan dari setiap studi kinematika adalah untuk mengembangkan model mental canggih yang berfungsi untuk menggambarkan (dan akhirnya, menjelaskan) gerakan objek dunia nyata.[2]

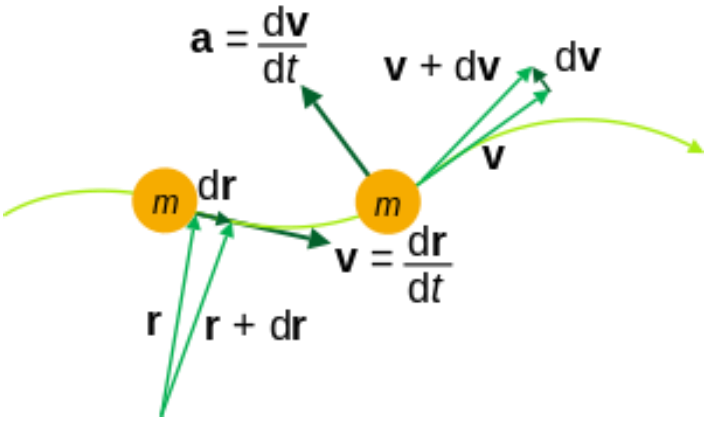

Figure 1: Kinematika Lintasan Partikel [3]

- Kecepatan (V)
  Menurut makna kecepatan, itu dapat didefinisikan sebagai laju perubahan posisi benda terhadap kerangka acuan dan waktu. Ini mungkin terdengar rumit tetapi kecepatan pada dasarnya melaju ke arah tertentu. Ini adalah besaran vektor, yang berarti kita membutuhkan besaran (kecepatan) dan arah untuk menentukan kecepatan. Satuan SI adalah meter per detik (ms-1). Jika ada perubahan besar atau arah kecepatan suatu benda, benda tersebut dikatakan mengalami percepatan.[4]

$$\frac{dx(t)}{dt} = v \Rightarrow \int \frac{dx(t)}{dt}\,dt = \int v\,dt \Rightarrow x(t) = v \times t$$

Figure 2: Rumus Kecepatan [5]

- Percepatan (a)
  Percepatan termasuk besaran vektor karena memiliki besar dan arah. Ini juga merupakan turunan kedua posisi terhadap waktu atau turunan pertama kecepatan terhadap waktu.[6]

$$\frac{d^2x(t)}{dt^2} = a \rightarrow \int \frac{d^2x(t)}{dt^2}dt = \int a\,dt \rightarrow \int \frac{dx(t)}{dt}dt = \int at\,dt$$

$$\frac{dx(t)}{dt} = at \qquad x(t) = \frac{1}{2} \times a \times t^2$$

Figure 3: Rumus Percepatan [5]

- Vektor dan Skalar
  Matematikawan dan ilmuwan menyebut besaran yang bergantung pada arah sebagai besaran vektor. Besaran yang tidak bergantung pada arah disebut besaran skalar. Besaran vektor memiliki dua sifat yaitu besaran dan arah. Besaran skalar hanya memiliki besaran. Saat membandingkan dua besaran vektor dengan jenis yang sama, Anda harus membandingkan besar dan arahnya. Untuk skalar, Anda hanya perlu membandingkan besarnya. Saat melakukan operasi matematika pada besaran vektor (seperti menambah, mengurangi, mengalikan ..) Anda harus mempertimbangkan besaran dan arahnya. Hal ini membuat berurusan dengan besaran vektor sedikit lebih rumit daripada skalar.[7]

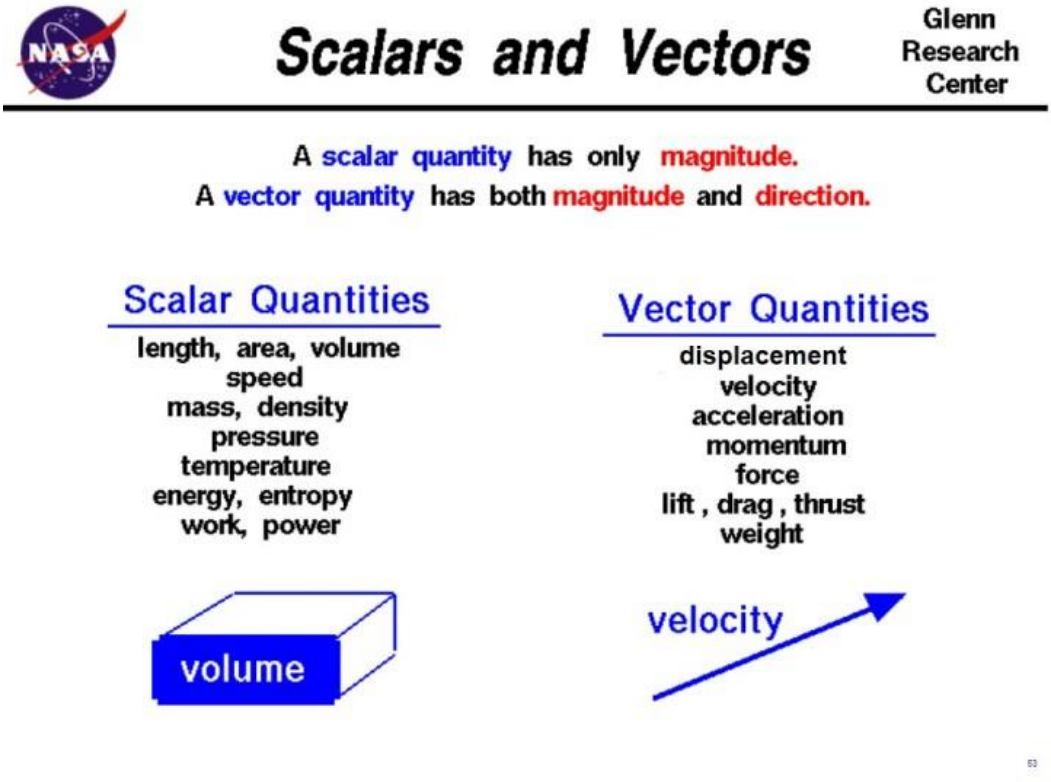

Figure 4: Perbedaan skalar dan vektor [7]

- Solusi Analitik dan Numerik
  Metode analitik menggunakan teorema eksak untuk menyajikan rumus dan memecahkan model. Sedangkan metode Numerik menggunakan prosedur pengulangan sederhana atau algoritma untuk memecahkan masalah dan juga metode Numerik mudah dibangun dengan bantuan komputer. [8]

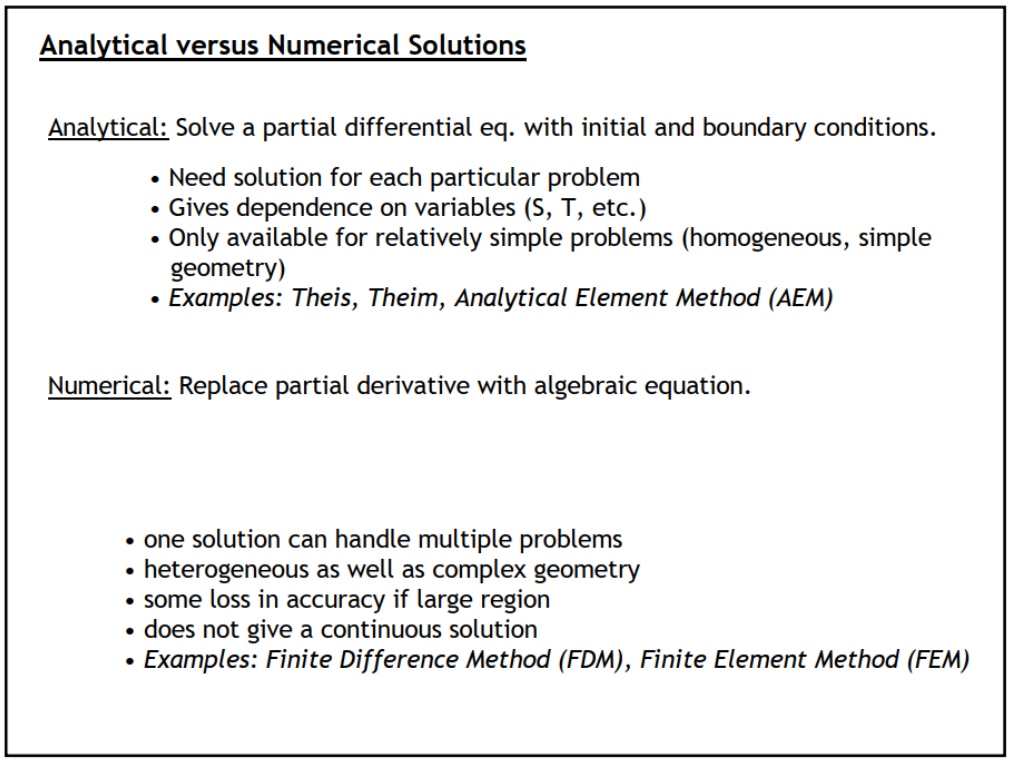

Figure 5: Perbedaan Solusi Analitik dan Numerik [9]

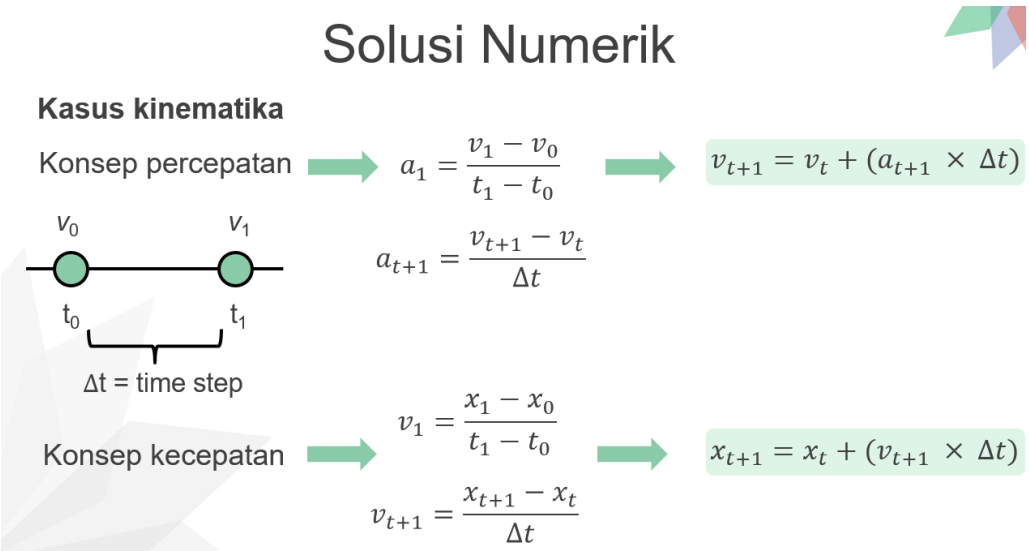

Figure 6: Solusi Numerik untuk Kecepatan dan Percepatan [5]

2.2 Gerak Peluru / Proyektil

Proyektil adalah benda apa pun yang dilemparkan ke luar angkasa di mana satu-satunya gaya yang bekerja adalah gravitasi. Gaya utama yang bekerja pada proyektil adalah gravitasi. Ini tidak berarti bahwa gaya lain tidak bekerja padanya, hanya saja efeknya minimal dibandingkan dengan gravitasi. Jalur yang diikuti oleh proyektil dikenal sebagai lintasan. Bola bisbol yang dipukul atau dilempar adalah contoh proyektil. Ketika sebuah partikel dilemparkan secara miring di dekat permukaan bumi, ia bergerak sepanjang jalur melengkung di bawah percepatan konstan yang diarahkan ke pusat bumi (kita asumsikan bahwa partikel itu tetap dekat dengan permukaan bumi). Lintasan partikel semacam itu disebut proyektil dan geraknya disebut **gerak proyektil**.[10]

Dalam Gerak Proyektil, ada dua gerakan bujursangkar independen simultan:[10]

1) **Sepanjang sumbu x:** kecepatan seragam, bertanggung jawab untuk gerakan horizontal (maju) partikel.
2) **Sepanjang sumbu y:** percepatan seragam, bertanggung jawab untuk gerakan vertikal (ke bawah) partikel.

Percepatan dalam gerak proyektil horizontal dan gerak proyektil vertikal suatu partikel: Ketika sebuah partikel diproyeksikan di udara dengan kecepatan tertentu, satu-satunya gaya yang bekerja padanya selama waktu itu di udara adalah percepatan gravitasi (g). Percepatan ini bekerja secara vertikal ke bawah. Tidak ada percepatan dalam arah horizontal, yang berarti bahwa kecepatan partikel dalam arah horizontal tetap konstan. Mari kita perhatikan sebuah bola yang diproyeksikan pada sudut θ terhadap sumbu x horizontal dengan kecepatan awal u seperti yang ditunjukkan di bawah ini:[10]

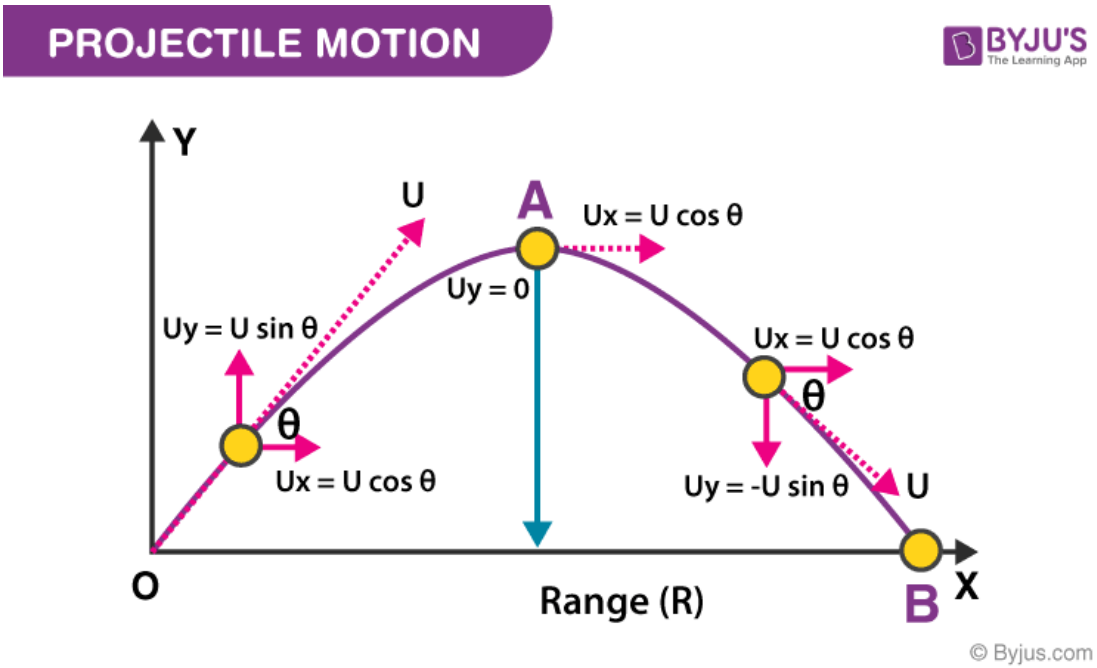

Figure 7: Gerak Proyeksi [10]

Titik O disebut **titik proyeksi**; **θ** adalah **sudut proyeksi** dan **OB** = ***Horizontal Range*** atau *Simply Range*. Total waktu yang dibutuhkan partikel dari mencapai O ke B disebut **waktu terbang.**[10]

Untuk menemukan parameter berbeda yang terkait dengan gerakan proyektil, kita dapat menggunakan **persamaan diferensial gerakan**:[10]

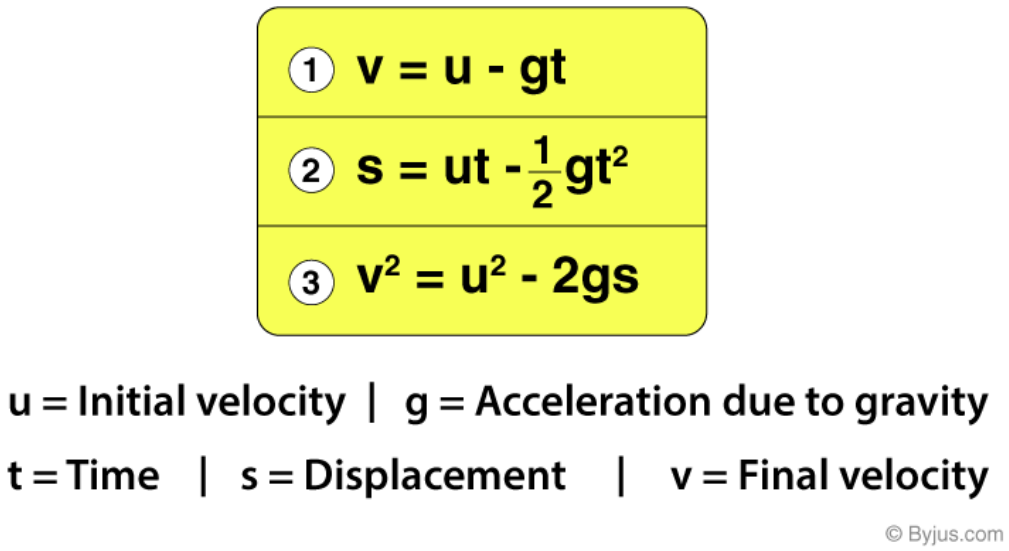

Figure 8: Persamaan Diferensial Gerakan [10]

Rumus solusi analitik untuk x(t) dan y(t) dan didefinisikan sebagai berikut:

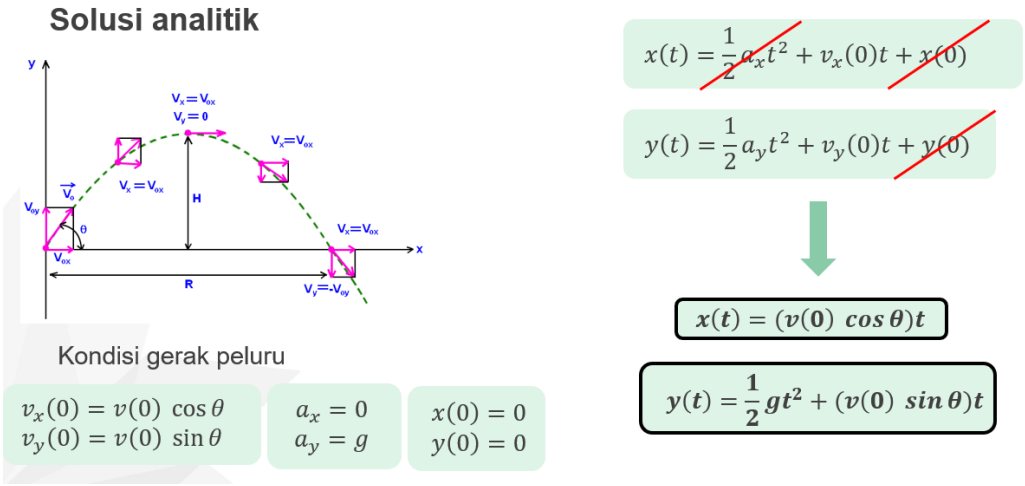

Figure 9: Rumus x(t) dan y(t) untuk solusi analitik [11]

Adapun untuk solusi numerik dapat didefinisikan sebagai:

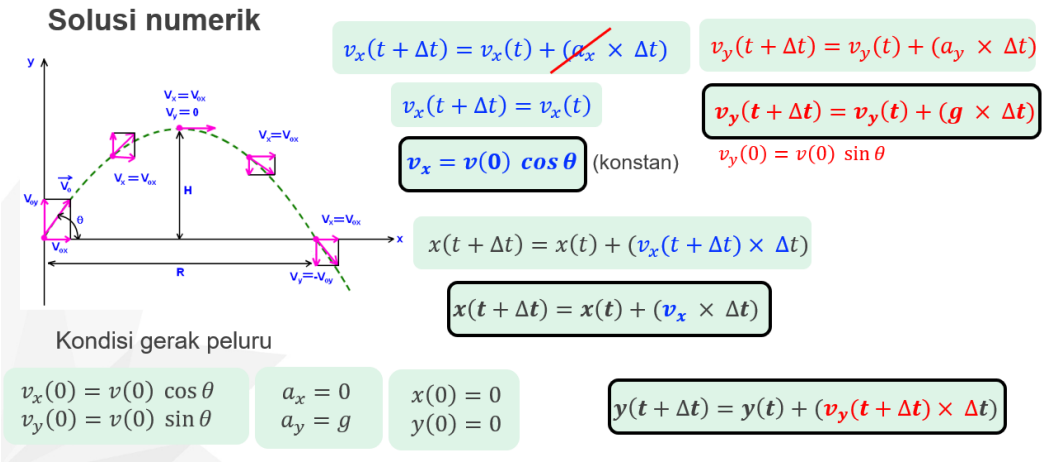

Figure 10: Rumus x(t) dan y(t) untuk solusi numerik - 1 [11]

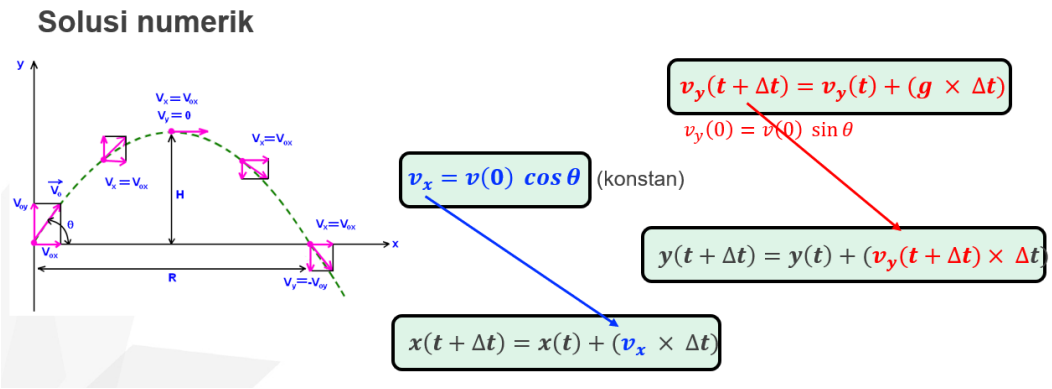

Figure 11: Rumus x(t) dan y(t) untuk solusi numerik – 2 [11]

<u>Total Waktu Terbang</u>: Perpindahan resultan (s) = 0 dalam arah Vertikal. Oleh karena itu, rumus waktu terbang diberikan dengan menggunakan Persamaan gerak:

gt2 = 2(uyt – sy) [Disini, uy = u sin θ dan sy = 0]

yaitu gt2 = 2t × u sin θ

Oleh karena itu, rumus waktu terbang (t) diberikan oleh:[10]

$$Total\ Time\ of\ Flight\ (t) = \frac{2u \sin \Theta}{g}$$

Figure 12: Rumus Total Waktu Terbang - 1 [10]

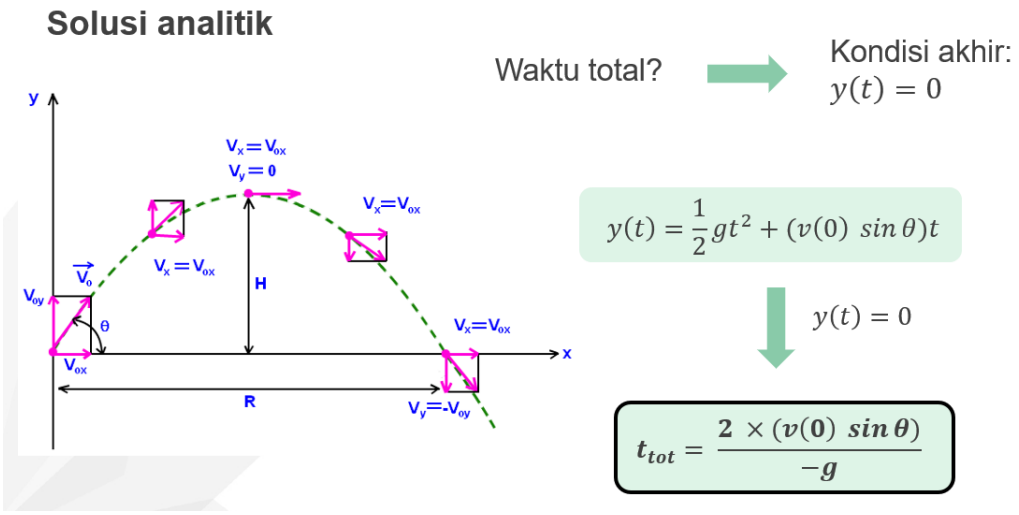

Figure 13: Rumus Total Waktu Terbang – 2 [11]

<u>Rentang Horizontal (OA)</u>: Komponen kecepatan horizontal (ux) × Total Waktu Terbang(t)

R = u cos θ × 2u×sinθg

Oleh karena itu, dalam gerakan proyektil Range Horizontal diberikan oleh (R):[10]

$$Horizontal\ Range\ (R) = \frac{u^2 \sin 2\Theta}{g}$$

Figure 14: Rumus Rentang Horizontal [10]

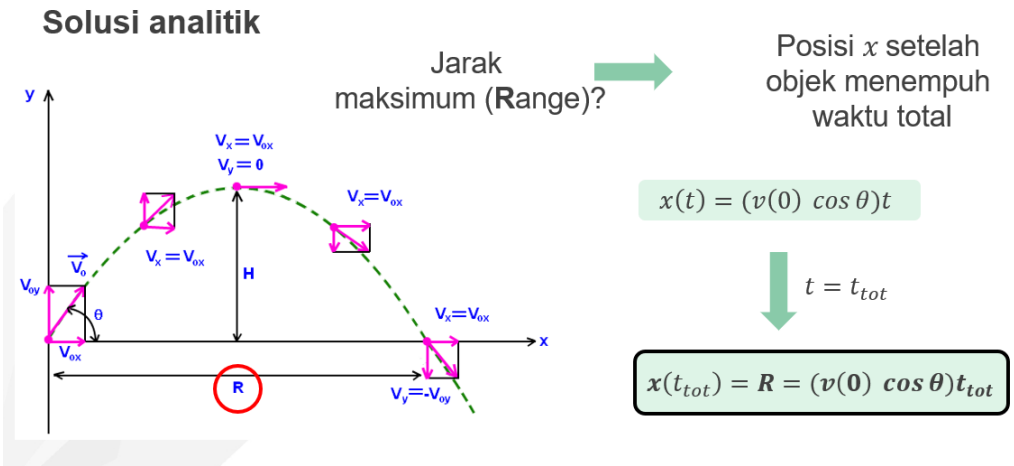

Figure 15: Rumus Jarak Maksimum [11]

Ketinggian Maksimum Proyektil: Setelah memahami apa itu proyektil dan apa itu gerak proyektil, beri tahu kami ketinggian maksimum proyektil. Ketinggian maksimum benda adalah posisi vertikal tertinggi sepanjang lintasannya. Perpindahan horizontal proyektil disebut jangkauan proyektil. Jangkauan peluru tergantung pada kecepatan awal benda.

Jika v adalah kecepatan awal, g = percepatan gravitasi dan H = ketinggian maksimum dalam meter, θ = sudut kecepatan awal dari bidang horizontal (radian atau derajat).

Ketinggian maksimum proyektil diberikan oleh rumus: [10]

$$H = \frac{v_0^2 sin^2\theta}{2g}$$

Figure 16: Rumus Ketinggian Maksimum Proyektil - 1 [10]

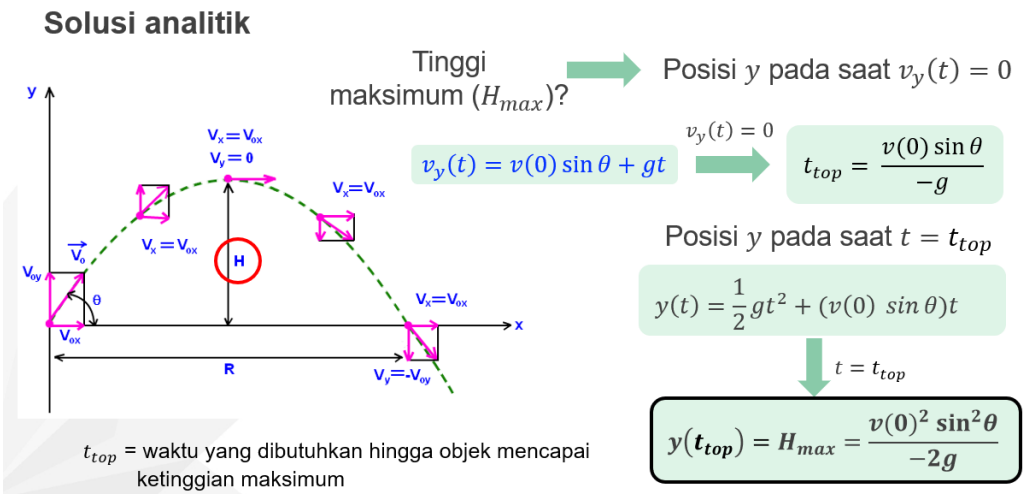

Figure 17: Rumus Ketinggian Maksimum Proyektil – 2 [11]

Persamaan Lintasan:

$$Equation\ of\ Trajectory = x\tan\Theta - \frac{gx^2}{2u^2cos^2\Theta}$$

Figure 18: Persamaan Lintasan [10]

Ini adalah persamaan lintasan pada gerak peluru, dan ini membuktikan bahwa gerak peluru selalu bersifat parabola.[10]

Pada kasus yang lebih riil, simulasi gerak peluru perlu mempertimbangkan faktor hambatan udara yang mempengaruhi pergerakan objek. Secara umum, perhitungan posisi objek pada gerak peluru dengan mengabaikan atau mempertimbangkan hambatan udara adalah sama. Perbedaan utama untuk kedua kasus tersebut hanyalahekspresi yang digunakan pada percepatan sumbu x dan y. Untuk kasus kedua,percepatan gravitasi pada sumbu dan diformulasikan oleh persamaan (11) dan (13), dimana D dan、m berturut-turut merepresentasikan konstanta dan massa objek.[1]

$$a_x = -\left(\frac{D}{m}\right) v v_x \quad (11)$$

$$a_y = -g - \left(\frac{D}{m}\right) v v_y \quad (12)$$

$$v = \sqrt{v_x^2 + v_y^2} \quad (13)$$

Figure 19: Rumus ax, ay dan v untuk gerak peluru dengan hambatan udara[1]

Dalam mencari gerak jatuh bebas dari kedua solusi diatas dapat dimplementasikan dengan algoritma, yang secara umum dapat digambarkan melalui *flowchart* sebagai berikut:

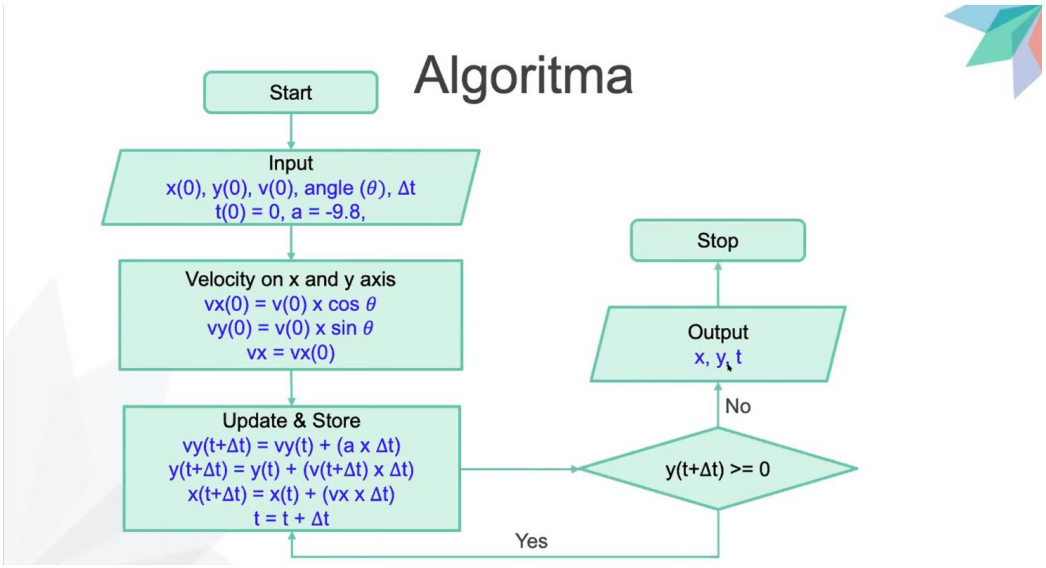

Figure 20: Flow Chart Algoritma Gerak Peluru [12]

Algoritma diatas dapat diimplementasikan menggunakan bahasa pemograman *Python* dengan bantuan beberapa library seperti *numpy* dan *matplotlib*. Pertama kita perlu mendefinisikan beberapa variabel untuk proses inisialisasi:

Tanpa Adanya Hambatan Udara

1. Inisiasi Variabel

```
[1]: import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns
     from math import sqrt

     # initialization
     x = 0                                  # initial x position
     y = 0                                  # initial y position
     v0 = 50                                # initial velocity
     angle = 35                             # initial angle
     angle_rad = (angle/360)*(2 * np.pi)    # convert degree to radian
     g = -9.806                             # gravity acceleration
     t = 0                                  # time
     dt = 0.01                              # time step
```

Figure 21: Kode Python dalam Proses Inisialisasi - 1 [13]

2. Inisiasi Array

```
[2]: # initialization array
     x_arr = [x]
     y_arr = [y]
     t_arr = [t]

     # velocity for x and y axis
     vx = v0 * np.cos(angle_rad)
     vy = v0 * np.sin(angle_rad)
```

Figure 22: Kode Python dalam Proses Inisialisasi – 2 [13]

Kedua, kita bisa langsung membuat implementasi untuk solusi analitik dan numerik:

- Kode *Python* untuk Solusi Numerik

3. Update

```
[4]: # update
     while y >= 0:
         vy += g*dt
         y += vy*dt
         x += vx*dt
         t += dt
         if y < 0:
             break
         # store
         x_arr.append(x)
         y_arr.append(y)
         t_arr.append(t)
```

Figure 23: Kode Python untuk Solusi Numerik - 1 [13]

4. Solusi Numerik

```
[5]: # numerical solution
     # total time
     t_tot_num = t_arr[-1]
     # range
     range_num = x_arr[-1]
     # max height
     h_max_num = np.max(y_arr)
```

Figure 24: Kode Python untuk Solusi Numerik -2 [13]

- Kode *Python* untuk Solusi Analitik

5. Solusi Analitik

[6]:
```python
# exact solution
x_ex_arr = [0]
y_ex_arr = [0]
for t in t_arr:
    x_ex = v0 * np.cos(angle_rad) * t
    y_ex = (0.5 * g * t**2) + (v0 * np.sin(angle_rad) * t)
    x_ex_arr.append(x_ex)
    y_ex_arr.append(y_ex)

# total time
t_tot_ex = (2 * v0 * np.sin(angle_rad))/-g
# range
range_ex = v0 * np.cos(angle_rad) * t_tot_ex
# max height
h_max_ex = (v0**2 * np.sin(angle_rad)**2) / (-2 * g)
```

Figure 25: Kode Python untuk Solusi Analitik[13]

- Kode *Python* untuk validasi posisi

6. Visualisasi Validasi Posisi

[7]:
```python
sns.set_theme(style="whitegrid")
plt.figure(figsize=(8,5))

sns.lineplot(x=x_arr, y=y_arr, color='g', label='numerical')
sns.lineplot(x=x_ex_arr, y=y_ex_arr, color='b', label='analytical')
plt.axhline(c='black')
plt.axvline(c='black')

plt.title('Perbandingan Solusi Analitik dengan Numerik')
plt.xlabel("Jarak (m)")
plt.ylabel("Tinggi (m)")
plt.legend()

plt.show()
```

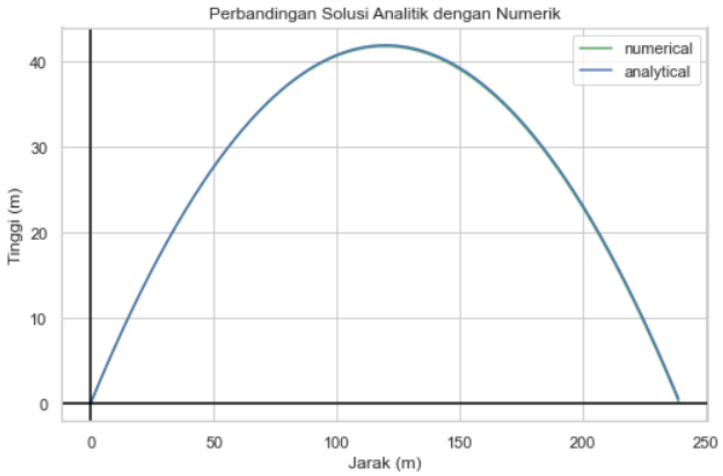

Figure 26: Kode Python untuk validasi posisi serta hasil visualisasi[13]

- Kode *Python* untuk validasi waktu

7. Validasi Waktu

[8]:
```
# compare
print('Numerik vs Analitik')
print('total waktu (s): {:.2f} vs {:.2f}'.format(t_tot_num, t_tot_ex))
print('Jarak (m) {:.2f} vs {:.2f}'.format(range_num, range_ex))
print('Tinggi (m) {:.2f} vs {:.2f}'.format(h_max_num, h_max_ex))
```

```
Numerik vs Analitik
total waktu (s): 5.83 vs 5.85
Jarak (m) 238.78 vs 239.57
Tinggi (m) 41.79 vs 41.94
```

Figure 27: Kode Python untuk validasi waktu dan hasilnya [12]

- Kode *Python* untuk simulasi

8. Pembuatan Simulasi

[ ]:
```
# plot for animation
plt.rcParams.update({'figure.max_open_warning': 0})
for i in range(len(y_arr)):
    plt.figure()
    plt.scatter(x_arr[i], y_arr[i], marker='o', c='b')
    plt.text(32, 14, '{:.2f} s'.format(t_arr[i]))
    plt.ylim(-1,16)
    plt.xlim(-1,36)
    plt.axhline(c='black')
    plt.axvline(c='black')
    plt.savefig('./images/fig_{:04d}.png'.format(i+1),
                format='png', dpi=1000, bbox_inches="tight")
```

Figure 28: Kode Python untuk simulasi [12]

Algoritma diatas jika mengalami hambatan udara dapat diimplementasikan menggunakan bahasa pemograman *Python* dengan bantuan beberapa library seperti *numpy* dan *matplotlib* juga. Pertama kita perlu mendefinisikan beberapa variabel untuk proses inisialisasi:

Dengan Hambatan Udara

1. Inisiasi Variabel Menggunakan Hambatan Udara

[9]:
```
# initialization
m = 0.15                            # Massa Benda
d = 0.0013                          # konstanta
x = 0                               # initial x position
y = 0                               # initial y position
v0 = 50                             # initial velocity
angle = 35                          # initial angle
angle_rad = (angle/360)*(2 * np.pi) # convert degree to radian
g = -9.806                          # gravity acceleration
t = 0                               # time
dt = 0.01                           # time step
```

Figure 29: Kode Python dalam Proses Inisialisasi - 1 [1], [13]

2. Inisiasi Array Menggunakan Hambatan Udara

```
[10]: # initialization array
      x_arr = [x]
      y_arr = [y]
      t_arr = [t]

      # velocity for x and y axis
      vx = v0 * np.cos(angle_rad)
      vy = v0 * np.sin(angle_rad)
```

Figure 30: Kode Python dalam Proses Inisialisasi – 2 [13]

Kedua, kita bisa langsung membuat implementasi untuk solusi numerik:

- Kode *Python* untuk Solusi Numerik

3. Update Menggunakan Hambatan Udara

```
[11]: # update
      while y >= 0:
          v = sqrt((vx**2) + (vy**2))
          ax = - (d/m) * v * vx
          ay = g - (d/m) * v *vy
          vy += ay*dt
          vx += ax*dt
          y += vy*dt
          x += vx*dt
          t += dt
          if y < 0:
              break
          # store
          x_arr.append(x)
          y_arr.append(y)
          t_arr.append(t)
```

Figure 31: Kode Python untuk Solusi Numerik - 1 [1], [13]

4. Solusi Numerik Menggunakan Hambatan Udara

```
[12]: # numerical solution
      # total time
      t_tot_num = t_arr[-1]
      # range
      range_num = x_arr[-1]
      # max height
      h_max_num = np.max(y_arr)
```

Figure 32: Kode Python untuk Solusi Numerik -2 [13]

- Kode *Python* untuk visualisasi validasi posisi

5. Visualisasi Validasi Posisi Menggunakan Hambatan Udara

```
[13]: sns.set_theme(style="whitegrid")
plt.figure(figsize=(8,5))

sns.lineplot(x=x_arr, y=y_arr, color='b', label='numerical')
plt.axhline(c='black')
plt.axvline(c='black')

plt.title('Perbandingan Solusi Menggunakan Hambatan Udara dengan Numerik')
plt.xlabel("Jarak (m)")
plt.ylabel("Tinggi (m)")
plt.legend()

plt.show()
```

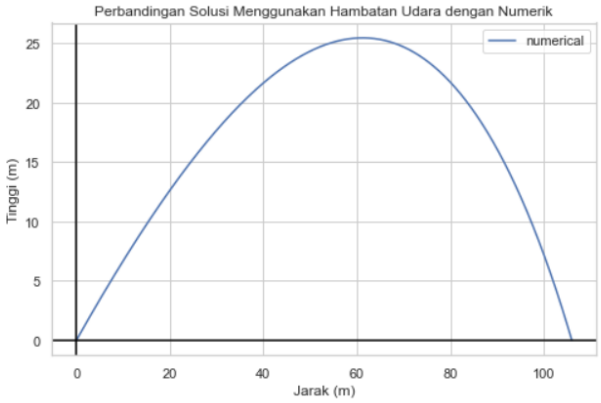

Figure 33: Kode Python untuk validasi posisi serta hasil visualisasi[13]

- Kode Python untuk validasi waktu

6. Validasi Waktu Menggunakan Hambatan Udara

```
[14]: print('Numerik Menggunakan Hambatan Udara')
print('total waktu (s): {:.2f}'.format(t_tot_num))
print('Jarak (m) {:.2f}'.format(range_num))
print('Tinggi (m) {:.2f}'.format(h_max_num))
```

```
Numerik Menggunakan Hambatan Udara
total waktu (s): 4.51
Jarak (m) 106.00
Tinggi (m) 25.45
```

Figure 34: Kode Python untuk validasi waktu dan hasilnya [12]

- Kode *Python* untuk simulasi

7. Pembuatan Simulasi Menggunakan Hambatan Udara

```
[ ]: # plot for animation
plt.rcParams.update({'figure.max_open_warning': 0})
for i in range(len(y_arr)):
    plt.figure()
    plt.scatter(x_arr[i], y_arr[i], marker='o', c='r')
    plt.text(32, 14, '{:.2f} s'.format(t_arr[i]))
    plt.ylim(-1,16)
    plt.xlim(-1,36)
    plt.axhline(c='black')
    plt.axvline(c='black')
    plt.savefig('./images/fig_{:04d}.png'.format(i+1),
                format='png', dpi=1000, bbox_inches="tight")
```

Figure 35: Kode Python untuk simulasi [12]

BAB III
PENUTUP

3.1 Kesimpulan

Penelitian atau percobaan diatas membuktikan untuk kasus pertama bahwa solusi numerik tanpa hambatan udara lebih lama durasinya, jaraknya lebih jauh dan lebih tinggi untuk ketinggian maksimum dari solusi numerik dengan hambatan udara, dimana suatu objek dengan massa 0.15 kg ditembakkan dari permukaan tanah dengan kecepatan awal 50 m/s dan sudut tembak 35° . Dengan menggunakan nilai D = 0.0013 dan Δt = 0.01, dimana g. adalah percepatan gravitasi yang nilainya -9.806 m/s^2.[1] Hasil yang didapat untuk solusi numerik tanpa hambatan udara adalah total waktu: 5.83 s, jarak: 238.78 m dan tinggi: 41.79 m. Adapun untuk solusi numerik dengan hambatan udara adalah total waktu: 4.51 s, jarak: 106.00 m dan tinggi: 25.45 m. Kemudian untuk kasus kedua berdasarkan percobaan diatas membuktikan bahwa solusi numerik hampir setara dengan solusi analitik yang perbedaannya sangat tidak terlalu signifikan dimana total waktu : 5.83 s (numerik) vs 5.85 s (analitik), jarak: 238.78 m (numerik) vs 239.57 m (analitik) dan tinggi: 41.79 m (numerik) vs 41.94 m (analitik).

Link GitHub:

https://github.com/dioapw/modeling-and-simulation-projectile-motion-final-project-1

3.2 Saran

Penyusunan laporan ini jauh dari kata sempurna. Oleh karena itu, berdasarkan kesimpulan diatas dapat diberikan saran, sebagai berikut:

- Menggunakan *time step* yang lebih kecil dibandingkan 0.01;
- Melakukan validasi analitik baik posisi maupun waktu untuk kasus hambatan udara;
- Memperbaiki pembuatan simulasi;
- Memperindah visualisasi.

Daftar Pustaka

[1] "tubes_1_Genap_2122".

[2] "Introduction to Kinematics," Mar. 2022. https://www.physicsclassroom.com/class/1DKin/Lesson-1/Introduction

[3] "Basics of Kinematics Boundless Physics," Mar. 2022. https://courses.lumenlearning.com/boundless-physics/chapter/basics-of-kinematics

[4] "Velocity - Definition, Units, Formula, Examples, Equations, Video and FAQs," Mar. 2021. https://byjus.com/physics/velocity

[5] D. Pemsis, "Gerak Jatuh Bebas Model."

[6] "What Is Acceleration - Formula, Unit, Examples, Types, FAQs," Mar. 2021, [Online]. Available: https://byjus.com/physics/acceleration

[7] "Scalars and Vectors." Mar. 2022. [Online]. Available: https://www.grc.nasa.gov/WWW/K-12/airplane/vectors.html

[8] "Numerical vs Analytical," Mar. 2016. https://cae4engineers.wordpress.com/2015/10/22/numerical-vs-analytical

[9] "Analytical versus Numerical Solutions Finite Difference Method."

[10] "Projectile Motion - Definition, Formula, Examples, Concepts, Video, and FAQs," Mar. 2021, [Online]. Available: https://byjus.com/physics/projectile-motion

[11] "Gerak Peluru Model Tim Dosen Pemodelan dan Simulasi."

[12] E. B. Setiawan, "Dr. Erwin Budi Setiawan - Algoritma dan Simulasi Gerak Peluru," Apr. 2021. https://www.youtube.com/watch?v=dSxbuA9Ut8Y

[13] T. D. Pemodelan and D. Simulasi, "Gerak Peluru Algoritma dan Simulasi."