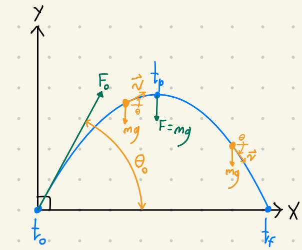
08/15/2025

Project 0 Theory: Projectile Motion



Assuming that the only external forces felt by the particle are that from gravity and of what propelled the particle into the air initially simplifies its model of analysis to what is sketched on the left.

This model marks the opening analysis for the quantification for numerous fields from archery and athletics to rocket science, video games, and targetting systems.

Before t_0: $F=F_0(t)$, launch

$\hat{F}_0=\cos\theta_0$

@ t_0: $\vec{v}=\dot{\vec{x}}=\dot{x}_0\hat{F}_0$, $\hat{x}_0=\hat{F}_0$

$x, y = 0, 0$

$\vec{F}=-mg\hat{j}$

Let $F_0, \theta_0, y_0, x_0, t_0, m, g$ = given

Assuming near Earth's surface,

$g=9.81\ (m/s^2)$

Let's model the below:

$\vec{F}(t)=F_x\hat{i}+F_y\hat{j}$ (N)

$\vec{a}(t)=\ddot{x}\hat{i}+\ddot{y}\hat{j}$ (m/s^2)

$\vec{v}(t)=\dot{x}\hat{i}+\dot{y}\hat{j}$ (m/s)

$\vec{r}(t)=x\hat{i}+y\hat{j}$ (m)

$\vec{F}(t)=0\hat{i}-mg\hat{j}$

$F=ma$

▶ $\vec{a}(t)=0\hat{i}-g\hat{j}$

$$\begin{cases}\ddot{x}=0\\ \ddot{y}=-g\end{cases}$$

If $\vec{a}$ = const. $\vec{v}$,

$\ddot{x}=\frac{d\dot{x}}{dt}=0$

▶ $\dot{x}=\dot{x}_0$

$v=\dot{x}$ @ $\dot{y}=0$

→ @ t_p

$\ddot{y}=-g=\frac{d\dot{y}}{dt}$

$d\dot{y}=-g\,dt$

$\int d\dot{y}=-\int g\,dt$

$\dot{y}_2-\dot{y}_1=-(gt_2-gt_1)$

$\Delta\dot{y}=g\cdot(t_1-t_2)$

$=\ddot{y}\cdot(t_2-t_1)$

▶ $\dot{y}=g(t_0-t)+\dot{y}_0$

$v_0\neq 0$

Recall Unit Circle

$\dot{x}_0=v_0\cos\theta_0$, $\dot{y}_0=v_0\sin\theta_0$

$\vec{v}_0=\dot{x}_0\hat{i}+\dot{y}_0\hat{j}$

→ Determined by launch $F_0 dt$

$F_0=ma_0$

$=m\frac{dv_0}{dt}$

$F_0 dt_0=m\,dv_0$, linear dy/dx

dv_0 = Launch v diff.

dt_0 = Launch t diff.

$=t_0-t_i$

→ Assume given v_0 for now

$\theta=\theta(t)$

$\vec{v}=v\cos\theta\hat{i}+v\sin\theta\hat{j}$

$=\dot{x}\hat{i}+\dot{y}\hat{j}$

$\dot{y}=g(t_0-t)+\dot{y}_0$

$\theta=\arctan\left(\frac{\dot{y}}{\dot{x}}\right)$ → From unit circle

$g(t_0-t)+\dot{y}_0=v\sin\theta$

$v=\frac{g(t_0-t)+\dot{y}_0}{\sin\theta}=\frac{\dot{y}}{\sin\theta}$

$\dot{x}$ = const.

$=v_0\cos\theta_0\overset{?}{=}v\cos\theta$

$a_0=0$ the instant $t=t_{now}=t_0$

$\dot{x}=\frac{dx}{dt}$, const.

$\int\dot{x}\,dt=\int dx$

$x_2-x_1=\dot{x}t_2-\dot{x}t_1$

$\Delta x=\dot{x}\Delta t$

$x=x(t)=\dot{x}(t-t_0)+x_0$

$=\int\dot{x}\,dt$

$\dot{y}=\frac{dy}{dt}$

$\int dy=\int\dot{y}\,dt$

$y_2-y_1=\int[g\cdot(t_0-t)+\dot{y}_0]\,dt=\overbrace{\int v\sin\theta\,dt}^{\text{useful if } \theta=\theta(t) \text{ is known/given/controlled}}$

$\Delta y=-\int a(\Delta t)dt+\int\dot{y}_0 dt$, $\dot{y}_0$ swaps signs @ $t=t_p$

$$\begin{cases}y_1=y(t<t_p)\\ \quad =\dot{y}_0\Delta t-\frac{1}{2}g(\Delta t)^2+y_0\\ y_p=y(t=t_p)\\ \quad =y_1=y_2\\ y_2=y(t>t_p)\\ \quad =y_1-\dot{y}_0\Delta t-\frac{1}{2}g(\Delta t)^2\end{cases}$$

$\Delta t_{02}=t_2-t_0$

$y_{max}=y_1(t=t_p)$

$=y_2(t=t_p)$

▶ Continuing analysis here is much quicker via simulations without defining any constraints. We can superpose our findings across both math (theory) & virtual test environments (like Octave) alike before purchasing materials & equipment for constructing prototypes.

08/19/2025

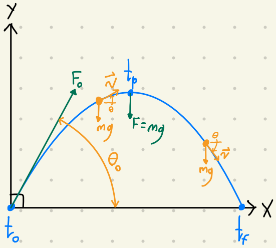
Before moving on to code (formalization), we need to define what we're plotting. While we have our functions (possible output parameters), we still need to define our input parameters and distinguish measured vs controlled inputs.

In simulation, time can be treated as a control and, in most cases, can be treated as a constant rate which is ideal for the independent (horizontal on graph) component for plotted data. The angle, programmed as either a function of time or as the function of a different function of time relating to the same particle, can work as both an output and measured input which is useful for validation.

This can be looped W.R.T. both time & theta without additional paper proofs.

08/19/2025

Project 0 Theory:
Programming (Formalizing) Projectile Motion
Ini Trial 0



<u>Initialize:</u>

V_Ø=1

t=0

theta_Ø = $\frac{\pi}{3}$ = 60°

theta = theta_Ø

g = 9.81

m = 1

$a = 0\hat{i} - g\hat{j}$

$V = V\_Ø * \cos(\text{theta\_Ø})\hat{i}$

$\quad + V\_Ø \sin(\text{theta\_Ø})\hat{j}$

$\quad - gt\hat{j}$

Review sequences & linspace(), numel(), length(), plot3()

<u>Update & Draw:</u>

```
for t=0.1:60
    plot
    axis
    drawnow;
    ~~pause(0.01);~~
end
hold off
```

>>help <topic>

google.com

1. Line styles
- `-` : Solid line (default)
- `--` : Dashed line
- `:` : Dotted line
- `-.` : Dash-dotted line
- `none` : No line (only markers)

2. Markers
- `o` : Circle
- `+` : Plus sign
- `*` : Asterisk
- `.` : Point
- `x` : Cross
- `_` : Horizontal line
- `|` : Vertical line
- `s` or `square` : Square
- `d` or `diamond` : Diamond
- `^` : Upward-pointing triangle
- `v` : Downward-pointing triangle
- `>` : Right-pointing triangle
- `<` : Left-pointing triangle
- `p` or `pentagram` : Pentagram
- `h` or `hexagram` : Hexagram

3. Colors
- Standard colors can be specified using a single character like:
  - `r` : Red
  - `g` : Green
  - `b` : Blue
  - `c` : Cyan
  - `m` : Magenta
  - `y` : Yellow
  - `k` : Black
  - `w` : White

4. Displayname
- `;displayname;` : Enclose the desired legend label in semicolons.

Example

octave
```
t = 0:0.1:10*pi;
plot3 (cos(2*pi*t), sin(2*pi*t), t, "r-.s;Helix;");
```

This example plots a red dash-dotted line with square markers and labels it "Helix" in the legend.

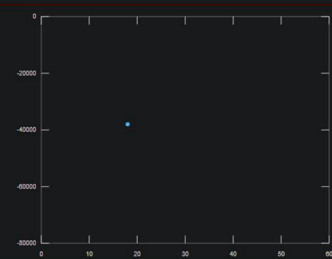
Function Reference: plot - Octave Forge
In this case, both matrices must have the same number of rows and...
Octave Forge

Function Reference: plot3 - Octave Forge
in which the arguments are taken to be the vertices of the points to be...
Octave Forge

AI responses may include mistakes. Learn more

Decaligo_081925-Project0_Ini0_Trial0.m

```
% Initialize:
v_0 = 1 %(m/s)
x_0 = 0 %(m)
y_0 = 0 %(m)
t = 0   %(s)
theta_0 = pi/3
theta = theta_0

% Properties
g = 9.81    %(m/s^2)
m = 1       %(kg)
dt = 2      %(s)
flag = 40/dt;

% Define Trajectory
a = [0
    -g]
v = [(v_0*cos(theta_0))
    (v_0*sin(theta_0)-g*t)]
r = [x_0
    y_0]

% Update vector functions v, r per step of time, a is const.
for i = t:flag                    % Loop once every dt
    plot(r(1),r(2))               % Plot r
    axis([0 60 -80000 10]);       % Disable auto axis scaling
    drawnow;                      % Force draw
    t += dt                       % Update current time
    r = [r(1) + (v(1)*dt)         % Update position
        r(2) + (v(2)*dt)]
    v = [v(1)                     % Update velocity
        v(2)-g*t]
endfor
```



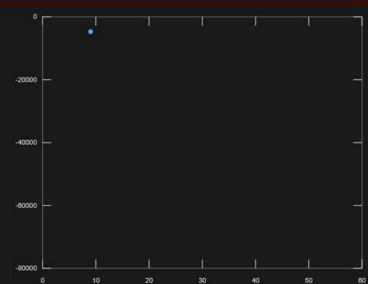
<< >>

Plot: Line 21

png svg

Vars

[2x1] a
\# ans
\# dt
\# flag
\# g
\# i
\# m
[2x1] r
\# t
\# theta
\# theta_0
[2x1] v
\# v_0
\# x_0
\# y_0

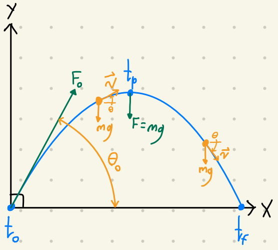


```
t = 20
r =

   10.000
 -6457.279

v =

   5.0000e-01
  -1.0782e+03
```

08/20/2025

Project 0 Theory: Programming (Formalizing) Projectile Motion

Ini Trial 1



08/25/2025

To Do: 0.0.0c

- Plot coordinate data on x,y graph as a function of time. Trial 0 only plotted one position per dt and there was a graph for each timestamp instead of all being plotted on the same graph.

Initialize:

V_0=1

dt=2

t=0:dt:60

theta_0 = $\frac{\pi}{3}$ = 60°

theta=theta_0

g=9.81

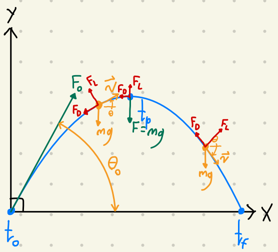
m=1

$a = 0\hat{i} - g\hat{j}$

V = V_0*cos(theta_0)$\hat{i}$ + V_0 sin(theta_0)$\hat{j}$ − g t $\hat{j}$

08/22/2025

Project 0 Initiation: Drafting Project Proposal & Abstract



Our reference point, established as the starting point for this traditional projectile motion trajectory due to a launch and the force of gravity, has been left open to design consideration. Furthermore, we intentionally neglected friction, drag, lift, and body mechanics to open up a much swifter introduction for analysis on the project so as to draft up a much more informed written abstract and starting WBS.

Objectives vs Goals:
Goals are typically very vague while objectives are more specific and provide the **scope**, **function**, and **measurement** for *design & experimentation*

Problem Statement

To quantify the specifics for aiming and launching projectiles for both my portfolio and to encourage others to conduct their own portfolio projects.

Objectives

1- Analyze an introductory model
2- Design & analyze various projectile launcher apparatuses
3- Design & analyze projectile bodies
4- Design & simulate Modular Prototype
5- Construct & test Prototype

Methods vs Procedures:
Methods analyze the why and how for the project while the procedure is a more specific step by step approach for how to conduct a specific experiment.

Models for Analysis:

- Lift and drag
- Kinematic differential equations
- Newton's Laws
- Work-energy
- Launch force types
- Force/Area
- State Space Models

Motivation

Comprehension of this experiment and its theory serves as an excellent segway into understanding the engineering, physics, and mathematics for designing and constructing new inventions that blur the lines between different branches of science.

Methodology

1- Theoretical analysis done on paper employing mathematics and hand drawn schematics
2- Theoretical research confirmed via university praised textbooks & academic engineering background
3- Program & simulate via GNU Octave
4- Define systems, subsystems, & properties, then create WBS
5- Reiterate steps 1-3 for each subsystem/module
6- Model & test everything in CAD, 3D printing prototypes as needed
7- Risk analysis & test plan
8- Prototype
9- Compile formal documentation
10- Present

Problem with solution → Paper math & hand sketches for building the solution → Computer math & simulate the solution → CAD & virtual prototype the solution → List all potential dangers, risks, and causes for failure → Test Safely?

Yes → Construct and test physical prototype

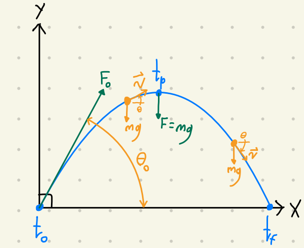
No → Problem with solution

Otherwise, or switch projects → List all potential dangers, risks, and causes for failure

08/22/2025

Project 0 Initiation: Defining the Systems

Initiation Model

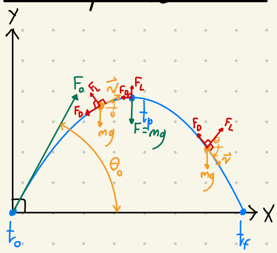


Initialize:

V_0=1
dt=2
t=0:dt:60
theta_0 = $\frac{\pi}{3}$ = 60°
theta=theta_0
g=9.81
m=1

$a = 0\hat{i} - g\hat{j}$
$V = V\_0 * \cos(theta\_0)\hat{i} + V\_0 \sin(theta\_0)\hat{j} - g t \hat{j}$

Aerodynamic Model:



New Parameters:

C_L=0 % Lift coef.
C_D=0 % Drag coef.

$$F_L = \frac{1}{2}\rho V^2 A C_L$$
$$F_D = \frac{1}{2}\rho v^2 A C_D$$

$$\hat{F}_L = \perp \hat{v}$$
$$\hat{F}_D = \parallel \hat{v}$$

Launch System:

Infinite possibilities → Categoriz. needed
* How are we propelling the projectile into the air?

Launcher Ideas:

- Portable vs mounted
- Thrusted into the air by a moving plate/rail
- Spring loaded (linear vs torsion)
- Elastic band (compare to spring loaded) i.e. crossbow, bow
- Velocity from centripetal motion (compare to thrusted into the air)
- Fluid pressure loaded (compare to spring loaded)
- Electromagnetic
- Solenoid (compare to simple electromagnet)

WARNING:
The launcher apparatus will likely inherit friction or other resistances to control and motion that will introduce error in the initial flight conditions if not accounted for.

The control system will need to be designed to take this in account if trajectory accuracy is of concern.

A Sensor System may be needed for developing a targeting control system that ensures a projectile hits a moving target

The Control System

- Can be human, mechanical, electrical, software, or a combination of such
- This is the system that selects the destination and/or trajectory inputs and ensures the launcher and its function follow through as intended
- If the launcher is hand held then it is almost certainly driven by a human for its core control system
- If not portable, will almost certainly be driven by a mechatronic control system driven by software. This is likely the most useful control system for this experiment

08/22/2025

Project 0 Initiation: WBS Ini. (Work Breakdown Schedule)

0 - Decaligo Consilideon: Project 0

- **0.0 - Project Initiation**
  - 0.0.0 - Projectile Motion Theory
    - 0.0.0a - Problem Statement
    - 0.0.0b - Ini. Trial 0
    - 0.0.0c - Ini. Trial 1
  - 0.0.1 - Abstract Draft
  - 0.0.2 - Define systems, subsystems, properties, etc
- **0.1 - Trajectory Mapping**
  - 0.1.0 - Finish Ini. Trial 1
  - 0.1.1 - Study Lift and Drag
  - 0.1.2 - Informal Analysis
  - 0.1.3 - Formal Analysis
- **0.2 - Projectile Body**
  - 0.2.0 - Review Fluid Mechanics
  - 0.2.1 - Study aerodynamic bodies
  - 0.2.2 - Informal analysis
  - 0.2.3 - Formal Analysis
- **0.3 - Launcher Apparatus**
  - 0.3.0 - Review Impulse-Momentum & Work-Energy Principles
  - 0.3.1 - Introduce Launch Actuation Types
  - 0.3.2 - Informal Analysis
  - 0.3.3 - Formal Analysis
- 0.4 - CAD
- 0.5 - Control System
- 0.6 - Documentation
- 0.7 - Prototype
- 0.8 - Presentation

08/26/2025:
The Sprint Board