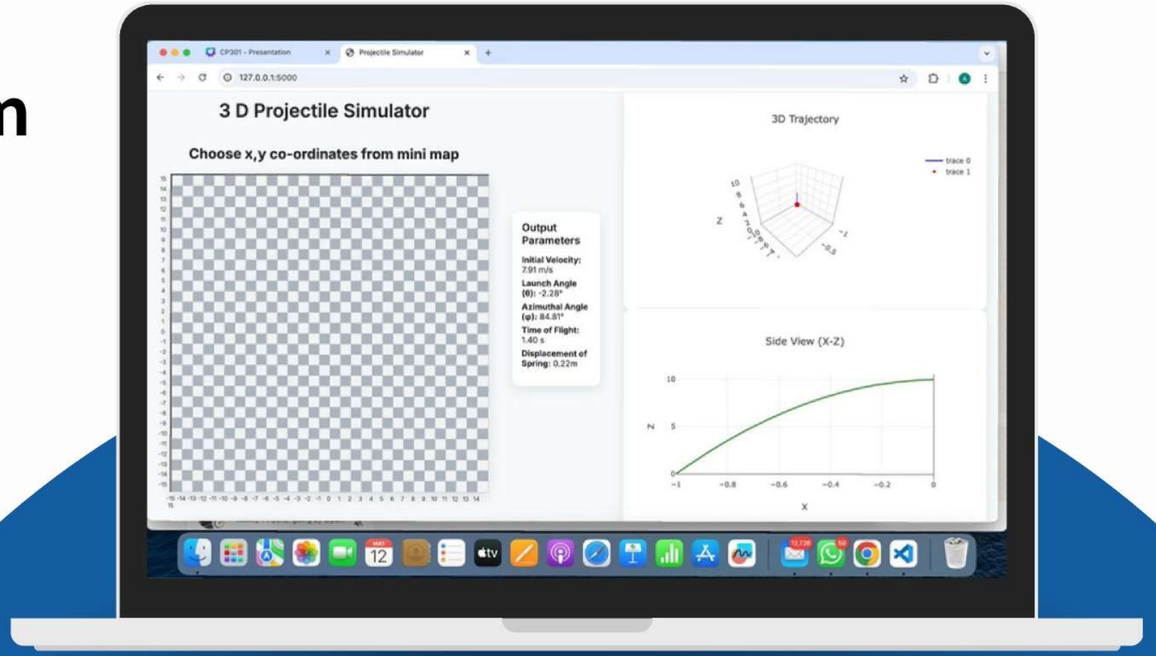
Department of
Mechanical Engineering
Indian Institute of Technology Ropar

CP301: Development Engineering Project

Fire Extinguisher Modelling
An Automated Fire Suppression System



Supervised by: Dr Chandrakant K. Nirala

Presented by:

Aditi - 2022MEB1288
Aditi Garg - 2022MEB1289
Arnav Maitreya - 2022MEB1299
Vaibhav Mehta - 2022MEB1360

Motivation and Overview

Fire remains one of the most hazardous events in both industrial and residential spaces. Traditional suppression methods such as sprinklers and manual extinguishers have limitations—sprinklers are static and slow, drones are complex and costly, and human-operated methods endanger lives.

There is a pressing need for an intelligent, quick-acting system that can detect and suppress fire remotely and autonomously. Our project addresses this gap by developing a smart system that combines AI-based fire detection with a spring-actuated launcher, delivering fire-suppressing projectiles accurately to the fire's location.

Objectives

Develop a robust mathematical model to simulate projectile motion in 3D and compute optimal launch parameters.

Design a spring-based launching mechanism and calculate required deflection to produce desired projectile velocity.

Implement a fire detection system using Convolutional Neural Networks (CNNs) for high accuracy.

Implement a fire detection system using ConvolutionalBuild an interactive web interface allowing users to simulate fire scenarios and test targeting algorithms..

Integrate all components into a unified system capable of automated fire detection and suppression.

System Overview

The system is an integrated fire response mechanism designed to autonomously detect and extinguish flames using precision-targeted projectiles. It brings together computer vision, mechanical control, and user interaction through a cohesive and intelligent design.

01

Fire Detection:
A camera-based CNN detects flames and localizes their coordinates in real-time. It ensures reliable detection across various environments.

02

Control Unit:
Calculates the optimal launch angle, velocity, and spring deflection. It adjusts dynamically based on the fire's position.

03

Launcher:
Uses a spring-mass mechanism to launch a nitrogen-filled ball. The ball bursts on impact to suppress the fire.

04

Feedback Loop:
Verifies that the spring is compressed correctly before launching. This improves targeting accuracy and safety.

05

Web Interface:
Users can input target points and view a 3D trajectory. It also provides basic system control and visualization.

Literature Review Summary

Mungan’s Projectile Optimization

Provided essential equations for determining minimum velocity and optimal angle for hitting targets at given coordinates.

Portable Water Mist Extinguishers

Offered insights into modern suppression methods and their efficiency across fire classes.

Fire Extinguishing Balls with Drones

Demonstrated feasibility and limitations of using nitrogen-based balls.

Educational Spring-Launch Setups

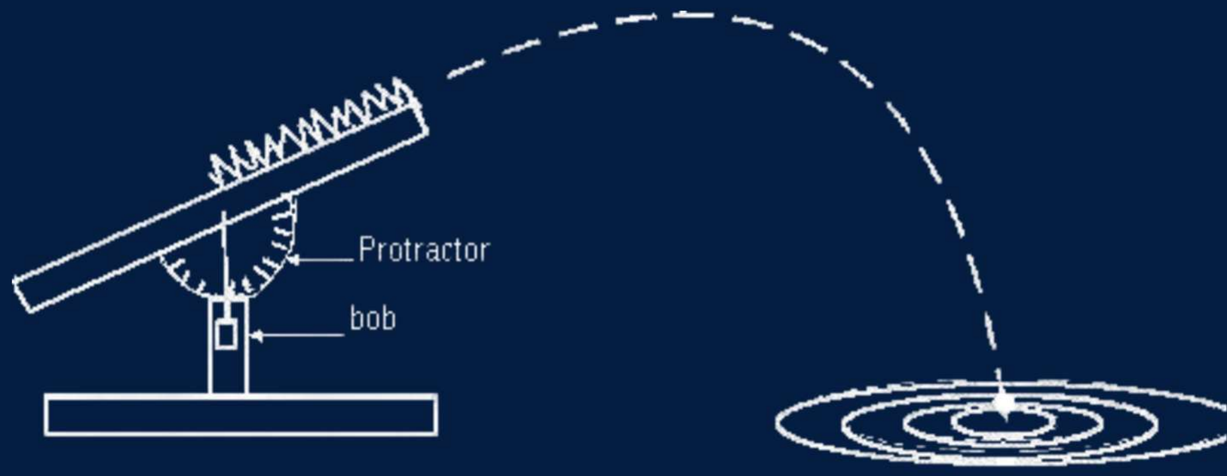
Validated our choice of spring mechanics and energy-based launch principles.

CNN-Based Fire Detection

Provided groundwork for designing a lightweight but accurate fire detection algorithm using AlexNet principles.

Methodology

- The room is modeled as a grid with known (x, y) coordinates to enable precise localization of fire.
- Temperature sensors and a CNN-powered camera are used to detect and verify the fire's location.
- These coordinates are fed into a mathematical model that calculates launch parameters using optimization algorithms.
- Spring compression is calculated from velocity using energy conservation ($\frac{1}{2}mv^2 = \frac{1}{2}kx^2$).
- A feedback system monitors spring compression and ensures exact displacement before launching the projectile.



Optimization & Projectile Calculation

We solve an optimization problem where the objective is to minimize the error between the projectile's landing point and the fire coordinates.

Parameters optimized:

Initial velocity (v_0)

Elevation angle (θ)

Azimuth angle (ϕ)

Time of Flight(t)

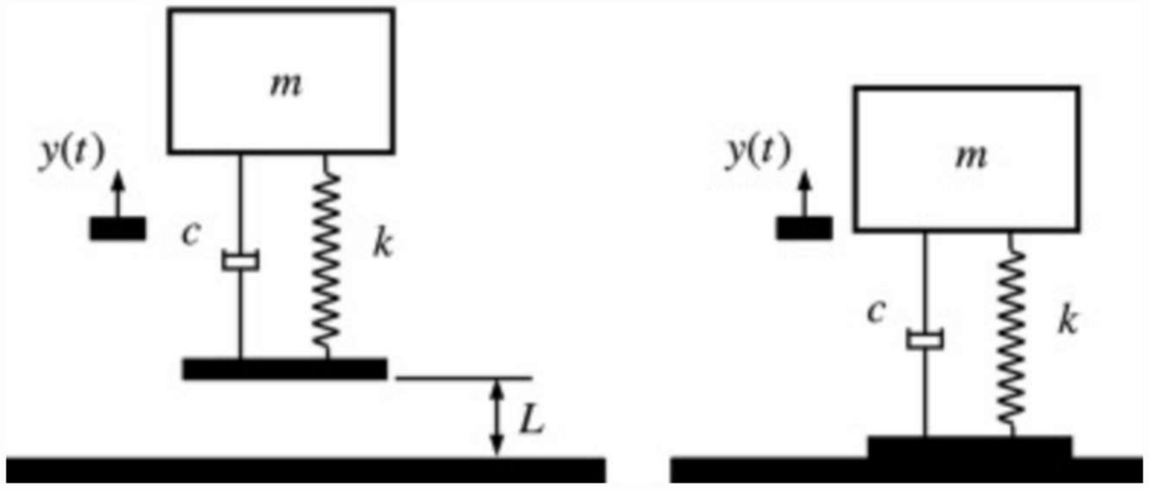
Displacement of Spring(x)

Vertical motion is modeled using kinematic equations, and solutions are validated against physical constraints (e.g., no imaginary roots, realistic time-of-flight). Optimization is performed using scipy.optimize.minimize with bounded constraints and initial guesses.

Spring Mass System

- Hooke's law is used to correlate the projectile's velocity to the spring's deflection:
  $\frac{1}{2}mv^2 = \frac{1}{2}kx^2 \Rightarrow x = \sqrt{mv^2/k}$
  Where:
- m = mass of projectile
- v = launch velocity from optimization
- k = spring constant (2000 N/m)

This mechanical calculation ensures the system's feasibility by matching spring energy with projectile kinetic energy.



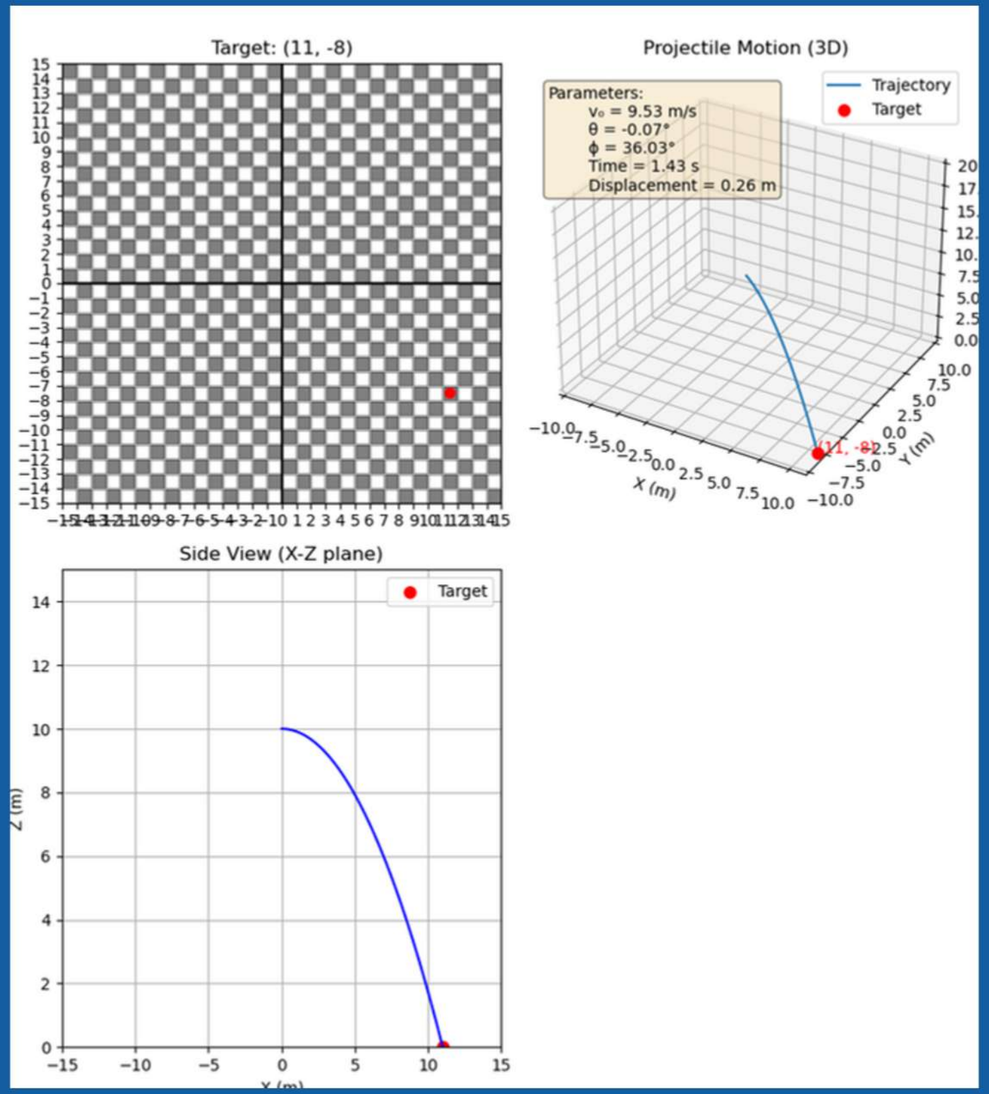
Visualization Interface

Users interact with a chessboard-style grid UI where clicking any square computes:

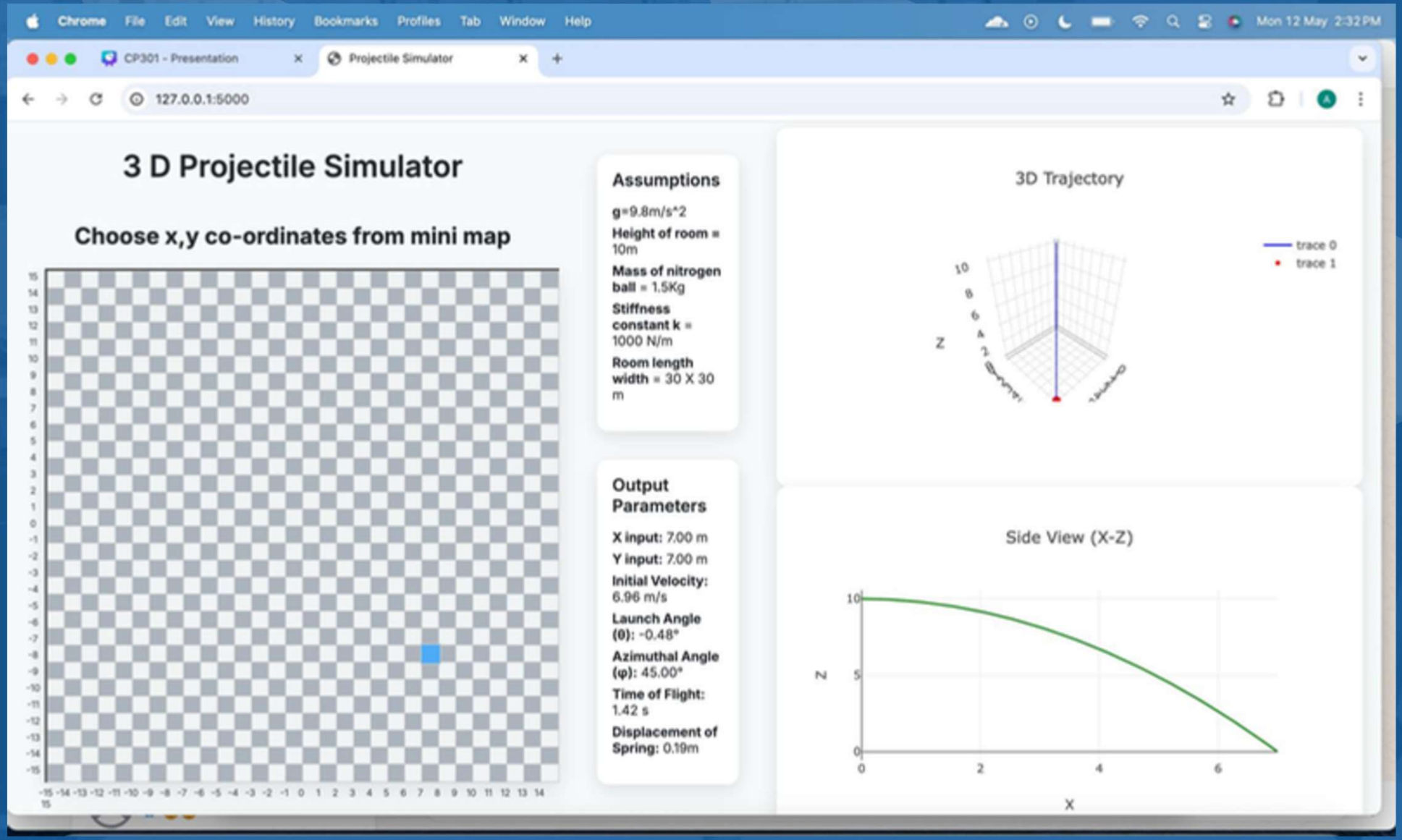
- Target coordinates (x, y)
- Optimized velocity, launch angle, azimuth
- Spring displacement

The resulting projectile path is visualized in 3D using Plotly and in 2D for side-view analysis. This interactive visualization bridges theory and real-world intuition, aiding both understanding and debugging.

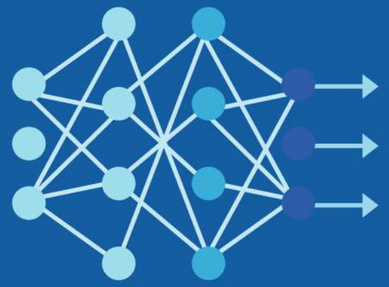
Responds in real-time (<1 sec) for user interaction.
Demonstrates strong modeling of energy conservation and projectile physics.



WEBSITE


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 12 May 2:32 PM
CP301 - Presentation
Projectile Simulator
127.0.0.1:5000
3 D Projectile Simulator
Choose x,y co-ordinates from mini map
Assumptions
g=9.8m/s^2
Height of room = 10m
Mass of nitrogen ball = 1.5Kg
Stiffness constant k = 1000 N/m
Room length width = 30 X 30 m
Output Parameters
X input: 7.00 m
Y input: 7.00 m
Initial Velocity: 6.96 m/s
Launch Angle (θ): -0.48°
Azimuthal Angle (φ): 45.00°
Time of Flight: 1.42 s
Displacement of Spring: 0.19m
3D Trajectory
trace 0
trace 1
Z
Side View (X-Z)
Z
X

Fire Detection – CNN Architecture



A custom 10-layer CNN was trained to distinguish fire vs. non-fire images:

- 3 Convolutional layers (32, 64, 128 filters)
- 3 MaxPooling layers
- 1 GlobalAveragePooling layer
- 1 Dense layer with 128 units
- 1 Dropout layer to prevent overfitting
- Final sigmoid output layer for binary classification
- Total parameters: ~110K
- Achieved ~96% classification accuracy

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 254, 254, 32)	896
max_pooling2d (MaxPooling2D)	(None, 127, 127, 32)	0
conv2d_1 (Conv2D)	(None, 125, 125, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 62, 62, 64)	0
conv2d_2 (Conv2D)	(None, 60, 60, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 30, 30, 128)	0
global_average_pooling2d (GlobalAveragePooling2D)	(None, 128)	0
dense (Dense)	(None, 128)	16,512
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129

Results – CNN Model

- **Training accuracy: ~100%**
- **Validation accuracy: ~95% (despite fluctuations)**

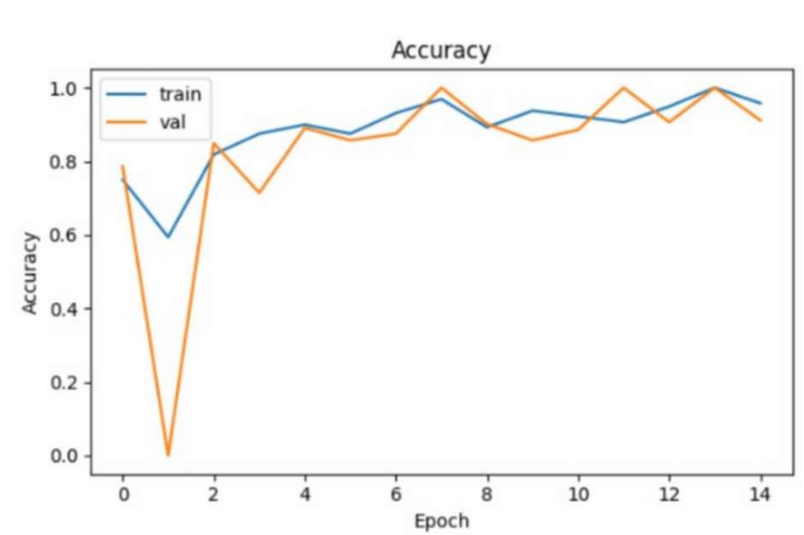
Confusion Matrix:

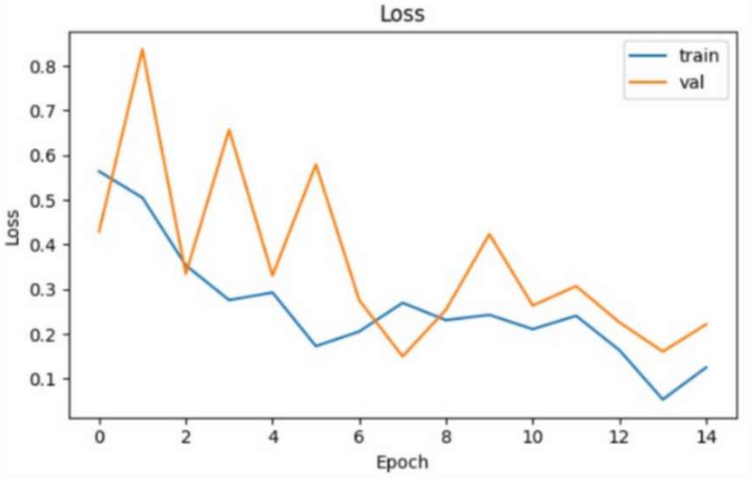
- **Fire: 144/151 correct (Recall: 0.95)**
- **No-Fire: 49/50 correct (Recall: 0.98)**

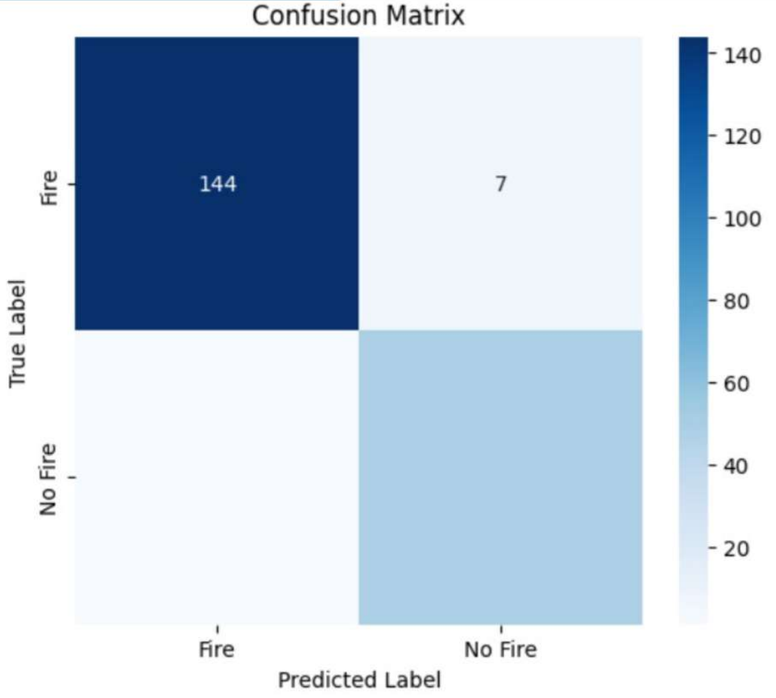
Test Accuracy: 95.83%

R^2 score: 0.8372 — Indicates good predictive performance and generalization

Robust model suitable for real-world fire detection. scenarios

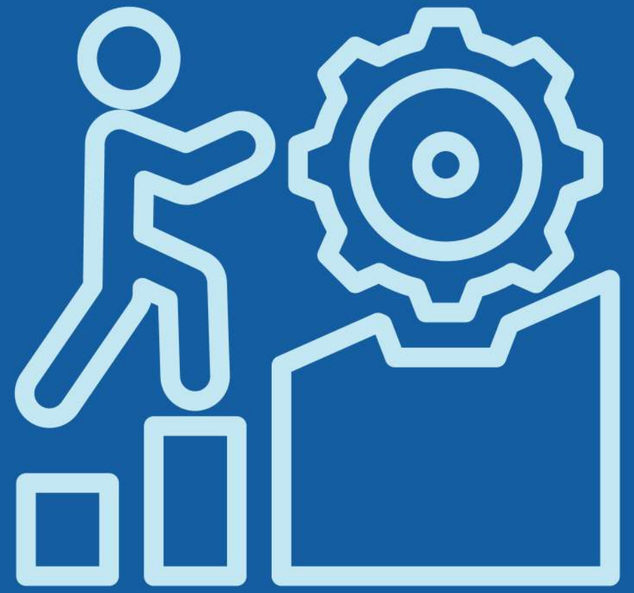






Challenges Faced

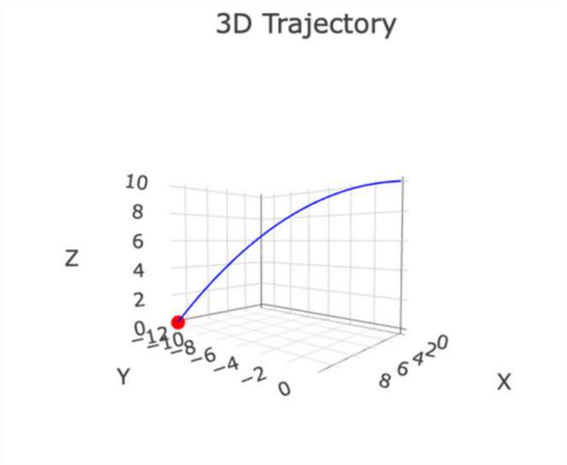
- Real-time optimization required efficient bounds and initialization
- Imaginary roots from projectile equations had to be filtered
- Multiple optimizations could get stuck in local minima
- Visualization axes needed syncing across 2D/3D views
- User input edge cases led to unreachable targets — handled via reflections and constraints



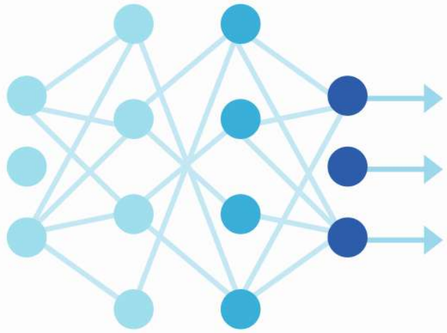
FUTURE WORK

- Hardware Implementation: Integrate IR sensors, thermal cameras, and physical spring launcher.
- Real-Time Motion Compensation: Add prediction algorithms (e.g., Kalman filters)
- IoT Integration: Cloud control, alert systems, historical data logging
- ML-Based Spread Prediction: Use data to forecast fire expansion and optimize targeting
- Multi-Launcher Architecture: Coordinate several launchers to cover zones
- Power Optimization: Solar-based systems and low-power electronics

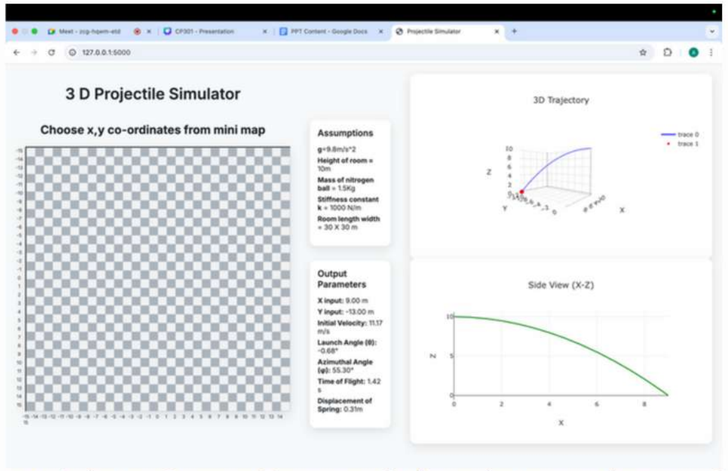
CONCLUSION



Simulated Projectile Motion matches Theory



CNN Model detects Fire reliably in Real-Time



Web interface offers real-time interaction and feedback

The platform is modular, scalable, and ready for real-world extensions, offering a novel solution to autonomous fire suppression in hazardous environments.

REFERENCES

- C. E. Mungan, “Optimizing the launch of a projectile to hit a target,” U.S. Naval Academy, Annapolis, MD.
- Z. Liu, A. Kim, and D. Carpenter, “A study of portable water mist fire extinguishers used for extinguishment of multiple fire types,” Fire Safety Journal, vol. 42, no. 1, pp. 25–42, Feb. 2007, doi: 10.1016/j.firesaf.2006.06.008.
- B. Aydin, E. Selvi, J. Tao, and M. J. Starek, “Use of fire-extinguishing balls for a conceptual system of drone-assisted wildfire fighting,” Department of Engineering & Technology, Texas A&M University- Commerce, Feb. 2019.
- A. A. Alsheikhy, "A Fire Detection Algorithm Using Convolutional Neural Network," JKAU: Engineering Sciences, vol. 32, no. 2, pp. 39–55, 2022, doi: 10.4197/Eng.32-2.3.
- P. K. Joshi, Z. Attaree, and P. K. Nawale, "Projectile motion," Homi Bhabha Centre for Science Education, TIFR, Mumbai, [Online].

THANK YOU!