Report on Projects for the IEEE Communications Society

A report on projects completed during the two-month break

July 31, 2025

Abstract

This report details the projects undertaken and completed for the IEEE Communications Society during the two-month break. The projects are designed to enhance practical skills in various domains, including software development with MATLAB, embedded systems programming with Arduino, and data visualization. This report covers the following projects: the MATLAB GUI-Based Calculator, a Bluetooth-based Home Automation system, and a MATLAB Weather Forecast Visualization project. The report also includes the MATLAB Onramp course completion, which was a prerequisite for these projects.

1 Introduction

This report documents the successful completion of a series of projects assigned by the IEEE Communications Society to be finished during the two-month break. The projects aimed to provide hands-on experience and develop skills in software programming, embedded systems, and data analysis. The foundational step for these projects was the completion of the MATLAB Onramp course on MathWorks, for which a completion certificate has been submitted. The following sections provide a detailed breakdown of each project, including problem statements, implementation details, and code snippets.

2 MATLAB Onramp Course Completion

As a prerequisite for the MATLAB-based projects, the MATLAB Onramp course on MathWorks was successfully completed. This course provided the fundamental skills necessary to work with MATLAB, including a basic understanding of the environment, data types, and scripting, which were directly applied in the subsequent projects.

3 Project 1: MATLAB GUI-Based Calculator

3.1 Problem Statement

The objective was to develop a simple, interactive calculator application using MATLAB, which provides a Graphical User Interface (GUI) to perform basic arithmetic operations—Addition, Subtraction, Multiplication, and Division—on two user-provided inputs.

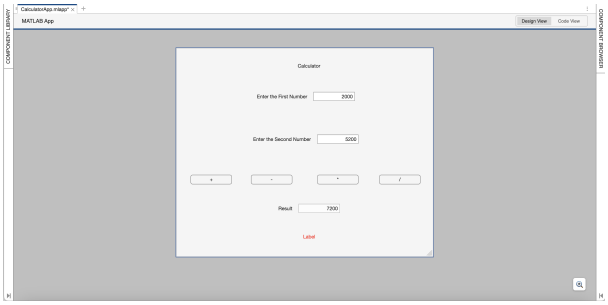
3.2 Project Requirements

- **Inputs**: Two numeric input fields for users to enter the numbers.
- **Operations**: The calculator should support the four basic arithmetic operations: Addition (+), Subtraction (−), Multiplication (×), and Division (÷).
- **Output**: A result display area to show the output of the operation.
- **User Interface**: A clean, minimal, and intuitive layout using MATLAB App Designer with clearly labeled buttons and fields.
- **Functionality**: When a user clicks an operation button, the application should read both input values, perform the selected operation, and display the result.
- **Constraints and Edge Cases**: The application should handle invalid inputs (non-numeric, empty fields) and division by zero gracefully, displaying a user-friendly error message. The UI should be responsive to all user actions.

3.3 Implementation Details and Code

The project was implemented using a command-line interface in MATLAB Online, as shown in the provided code. The code prompts the user for two numbers and an operation. A `switch` statement is used to perform the chosen arithmetic calculation. Edge cases, such as division by zero, are handled with an `if` statement to prevent a crash and display an appropriate error message.

```
% Basic Calculator in MATLAB Online
% --- Input Phase ---
% Get the first number from the user
num1 = input('Enter the first number: ');
% Get the operation from the user
operation = input('Enter the operation (+, -, *, /): ', 's'); % 's' reads input as a string
% Get the second number from the user
num2 = input('Enter the second number: ');
% --- Calculation Phase ---
result = 0; % Initialize result variable
switch operation
    case '+'
        result = num1 + num2;
        fprintf('Result: %.2f + %.2f = %.2f\n', num1, num2, result);
    case '-'
        result = num1 - num2;
        fprintf('Result: %.2f - %.2f = %.2f\n', num1, num2, result);
    case '*'
        result = num1 * num2;
        fprintf('Result: %.2f * %.2f = %.2f\n', num1, num2, result);
    case '/'
        % Handle division by zero
        if num2 == 0
            disp('Error: Division by zero is not allowed.');
        else
            result = num1 / num2;
            fprintf('Result: %.2f / %.2f = %.2f\n', num1, num2, result);
        end
    otherwise
        disp('Error: Invalid operation. Please use +, -, *, or /.');
end
```



4 Project 2: Bluetooth-based Home Automation

4.1 Project Title and Objective

The project, "Bluetooth-based Home Automation," aimed to create a system that allows users to control home appliances wirelessly using a smartphone via Bluetooth.

4.2 Development Environment and Tools

- **Bluetooth Module**: Simulated Bluetooth components
- **Development Environment**: Arduino IDE
- **Mobile Application Tool**: MIT App Inventor
- **Microcontroller Board**: Arduino UNO (simulated)

Note: This project was completed using online simulation platforms, Wokwi and Tinkercad, where standard components were used to replicate the functionality of a physical HC-05 module and Arduino UNO board.

4.3 Implementation Details

The project involved three main components: the Arduino hardware setup, the Arduino C++ code, and the Android application created with MIT App Inventor. The hardware and microcontroller were simulated on the online platforms. The core logic involved writing an Arduino sketch to read serial data from the simulated Bluetooth module and perform actions (e.g., toggling a digital pin to control a simulated LED or relay) based on the received commands. The MIT App Inventor application was designed to send specific characters via Bluetooth, which the Arduino then processed.

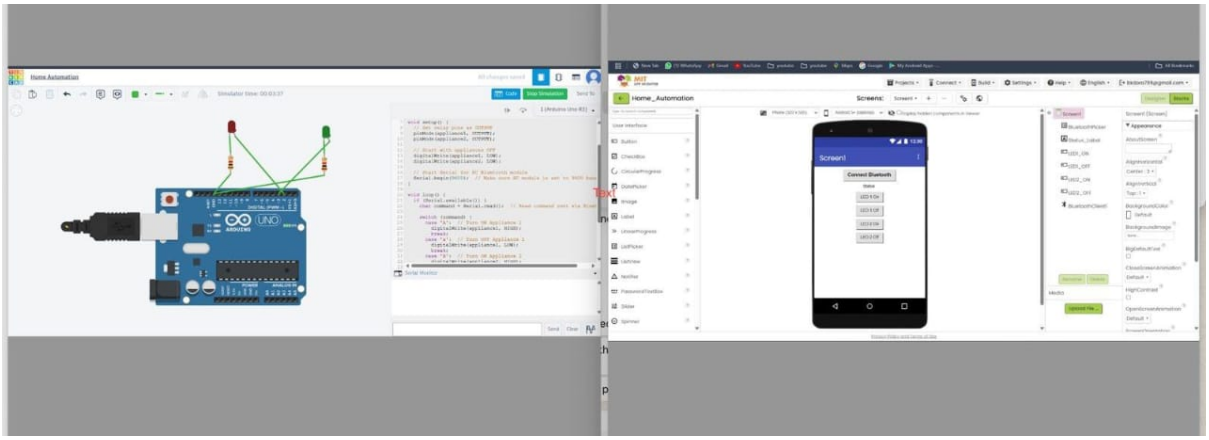
4.4 Arduino Code Snippet (Conceptual)

```
#include <SoftwareSerial.h>
SoftwareSerial bluetooth(10, 11); // RX, TX pins

char data = 0;
int relayPin = 7;

void setup() {
    Serial.begin(9600);
    bluetooth.begin(9600);
    pinMode(relayPin, OUTPUT);
    digitalWrite(relayPin, LOW); // Initially off
}

void loop() {
    if (bluetooth.available()) {
        data = bluetooth.read();
        if (data == '1') {
            digitalWrite(relayPin, HIGH); // Turn on
            Serial.println("Light ON");
```



```
        } else if (data == '0') {
            digitalWrite(relayPin, LOW); // Turn off
            Serial.println("Light OFF");
        }
    }
}
```

5 Project 3: MATLAB Project: Weather Forecast Visualization

5.1 Objective

The objective was to fetch historical weather data and plot trends like temperature and humidity to demonstrate skills in time-series plotting and data analysis.

5.2 Tools

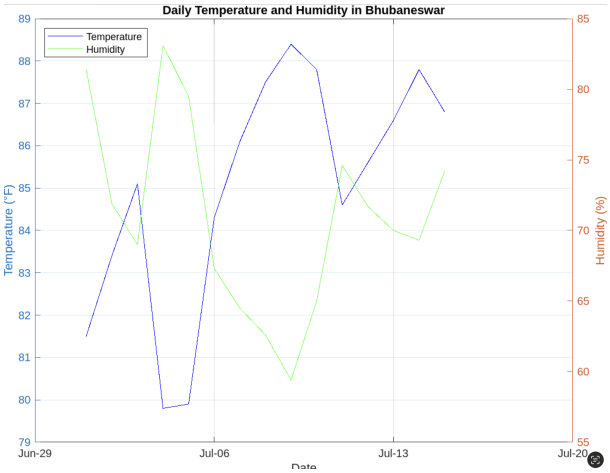
- **Skills**: Time-series plotting, data analysis.
- **Tools**: `datetime`, `plot`, `readtable`.

5.3 Implementation Details and Code

The project involved programmatically downloading historical weather data from the Visual Crossing API using `websave`, reading this data into a MATLAB table with `readtable`, and then preprocessing it. The preprocessing steps involved manually converting date and temperature/humidity columns to their correct data types (`datetime` and `numeric`) and handling missing values. Finally, the processed data was visualized using `plot` functions, creating separate plots for temperature and humidity, as well as a combined dual-y-axis plot for a comprehensive view. Summary statistics were also calculated and displayed.

5.4 MATLAB Code Snippet

```
% weather_visualization.m
% ... (Code for Data Acquisition and Preprocessing) ...
%% 3. Data Visualization
% Plotting Temperature over time
figure; % Create a new figure window
plot(T.Date, T.Temperature, 'b-o', 'LineWidth', 1.5, 'MarkerSize', 4);
title('Daily Temperature in Bhubaneswar');
xlabel('Date');
ylabel('Temperature (°F)'); % Assuming temperature is in Fahrenheit
grid on;
```




```
datetick('x', 'mmm-dd'); % Format x-axis ticks to show Month-Day
% Plotting Humidity over time
figure; % Create another new figure window
plot(T.Date, T.Humidity, 'g-s', 'LineWidth', 1.5, 'MarkerSize', 4);
title('Daily Humidity in Bhubaneswar');
xlabel('Date');
ylabel('Humidity (%)');
grid on;
datetick('x', 'mmm-dd'); % Format x-axis ticks to show Month-Day
% You can also create a combined plot if preferred
figure;
yyaxis left; % Use left y-axis for Temperature
plot(T.Date, T.Temperature, 'b-', 'DisplayName', 'Temperature');
ylabel('Temperature (°F)');
yyaxis right; % Use right y-axis for Humidity
plot(T.Date, T.Humidity, 'g-', 'DisplayName', 'Humidity');
ylabel('Humidity (%)');
title('Daily Temperature and Humidity in Bhubaneswar');
xlabel('Date');
grid on;
datetick('x', 'mmm-dd');
legend('show', 'Location', 'northwest');
% ... (Code for Summary Statistics) ...
```

6 Conclusion

The projects completed during the two-month break provided a comprehensive and practical learning experience across different technological domains. The MATLAB projects enhanced skills in GUI development, scripting, and data visualization, while the Bluetooth-based home automation project provided valuable hands-on experience with embedded systems and wireless communication in a simulated environment. The successful completion of these projects and the MATLAB Onramp course demonstrates a strong foundation in these technical areas, which will be beneficial for future endeavors within the IEEE Communications Society.