Convex Hull Algorithms Visualization and Performance Test

https://github.com/user-attachments/assets/cdfe3fe2-9a5c-4c40-bab2-2e60a3b0f473

Introduction

This project implements and visualizes various convex hull algorithms, specifically:

- **QuickHull**
- **Jarvis March**
- **Divide & Conquer**

The program provides both a graphical user interface (GUI) for visualization and a command-line interface (CLI) for performance testing.

Features

- **Algorithm Visualization**: Step-by-step visualization of convex hull algorithms for educational purposes.
- **CLI Mode**: Run algorithms on large datasets for performance measurement.
- **Data Generation**: Built-in functions to generate test datasets with various point distributions.
- **Automated Testing**: Scripts to execute algorithms on test datasets and log the results.

Installation

To build and run the program, you need:

- **C++ Compiler**: GCC with MinGW-w64 for Windows. Use this specific version to avoid compatibility issues with SFML.
- **CMake**: For building the project.
- **SFML Library**: For GUI visualization.

Building the Project

1. **Clone the Repository**:

```
git clone https://github.com/HackXIt/APRG-group-projects.git
```

2. **Run CMake Configuration:**

```
# Optional: Create a build directory
mkdir build && cd build
cmake ..
```

3. **Navigate to the Sub-Project inside the build directory:**

```
cd convex-hull
```

4. **Build the Project:**

4. **Build the Project:**

```
cmake --build .
```

The executable will be located in the `bin` directory.

Usage

The program can be run in two modes:

- **GUI Mode**: For visualization *(limited to a maximum of **50 data points**)*.
- **CLI Mode**: For performance testing on larger datasets.

Command-Line Interface (CLI)

The CLI allows you to execute the convex hull algorithms on specified or generated datasets.

Syntax

```
convex-hull.exe [OPTIONS]
```

Options

- `-h`, `--help`: Print help message.
- `-g`, `--gui`: Run with visualization using pre-loaded or generated data (limited to less than 50 points).
- `-a`, `--algorithm MODE`: Algorithm to use.
  - `0`: QuickHull
  - `1`: Jarvis March
  - `2`: Divide & Conquer
- **Data Input Options** *(Mutually Exclusive)*:
  - `-d`, `--data_file FILEPATH`: Path to a file containing points to load.
  - `-t`, `--test CASE`: Generate test data for the specified test case.
    - Valid test cases: `0` to `4`.
  - `-n`, `--number N`: Number of points to generate for the test case (used with `-t`). Default is `100`.

Note: You must specify either `--data_file` or `--test` when running in CLI mode unless you are running the GUI without pre-loaded data.

Data Input Options Details

- `--data_file FILEPATH`: Load data points from a file.
  - **File Format**:
    - **First Line**: Contains the number of points (integer).
    - **Subsequent Lines**: Each line contains the X and Y coordinates of a point, space-separated (floating-point values).
- `--test CASE`: Generate data points for a predefined test case.

    - **Test Cases**:
        - `0`: Random distribution of points.
        - `1`: Points forming a straight line.
        - `2`: Points forming a circle.
        - `3`: Random distribution inside a square (convex hull forms a square).
        - `4`: Large dataset with 100 million data points.
- `--number N`: Specify the number of points to generate for the test case (used with `--test`).

Examples

- **Run QuickHull Algorithm on Data File**:

```
convex-hull.exe -a 0 -d "path/to/data_file.txt"
```

- **Generate Test Case 2 (Circle) with 500 Points and Run Jarvis March Algorithm**:

```
convex-hull.exe -a 1 -t 2 -n 500
```

- **Run GUI with QuickHull Algorithm and No Pre-loaded Data**:

```
convex-hull.exe -g -a 0
```

- **Run GUI with Generated Test Case 1 (Straight Line) Data**:

```
convex-hull.exe -g -a 0 -t 1 -n 30
```

Important Notes

- **Mutual Exclusivity**:
    - The options `--data_file` (`-d`) and `--test` (`-t`) are mutually exclusive. You cannot use them together.
    - If neither `--data_file` nor `--test` is provided, the program will run the GUI without pre-loaded data (you can add points manually).
- **Visualization Limit**:
    - The GUI mode is limited to datasets with fewer than **50 points**.
    - If your dataset exceeds this limit, use CLI mode for performance testing.

Graphical User Interface (GUI)

The GUI provides a step-by-step visualization of the convex hull algorithms.

Running in GUI Mode

```
convex-hull.exe -g -a 0
```

You can optionally provide data using `--data_file` or generate data using `--test` and `--number`:

- **With Data File**:

```
convex-hull.exe -g -a 0 -d "path/to/data_file.txt"
```

- **With Generated Test Data**:

```
convex-hull.exe -g -a 0 -t 0 -n 40
```

Note: The dataset must contain fewer than **50 points** for visualization.

Controls

- `SPACE`: Step through the algorithm with visual explanation.
- `R`: Reset the visualization.
- `ENTER`: Run the algorithm to completion. *(1)*
- `Click in UI`: Place points manually. *(1)*

(1) Only possible if calculation has not started yet. If calculation already started, a reset is required.

Test Data Generation

The program includes built-in functions to generate test datasets with various point distributions using the `--test` and `--number` options.

Available Test Cases

- **Test Case 0**: Random distribution of points.
- **Test Case 1**: Points forming a straight line.
- **Test Case 2**: Points forming a circle.
- **Test Case 3**: Random distribution inside a square (convex hull forms a square).
- **Test Case 4**: Large dataset with 100 million data points.

Generating Test Data Example

- **Generate 1000 Random Points (Test Case 0) and Run QuickHull Algorithm**:

```
convex-hull.exe -a 0 -t 0 -n 1000
```

- **Generate 100 Million Points (Test Case 4) and Run Divide & Conquer Algorithm**:

```
convex-hull.exe -a 2 -t 4 -n 100000000
```

Note: Generating large datasets may require significant system resources.

Automated Testing

Automated testing is facilitated using a PowerShell script.

Test Script

PowerShell Script: `run_tests.ps1`

The PowerShell script executes the compiled binary with generated test data and logs the results.

Usage

```
.\run_tests.ps1 -BinaryPath "C:\path\to\convex-hull.exe" -LogFile "C:\path\to\test.log" # Optional: Log file path
```

Script Overview

- **Parameters**:
  - `-BinaryPath`: Full path to the `convex-hull.exe` executable.
- **Functionality**:
  - Iterates over all specified test cases and algorithms.
  - Executes the convex hull algorithms (`0`: QuickHull, `1`: Jarvis March, `2`: Divide & Conquer) on each test case.
  - Logs the output and errors to `test.log`.

Example Execution

```
# Navigate to the directory containing the script
cd C:\path\to\scripts

# Run the test script
.\run_tests.ps1 -BinaryPath "C:\path\to\convex-hull.exe"
```

O Notation

Jarvis March

- Best-Case: Line or Square Hull O(N)
- Average-Case: O(M*N)
- Worst-Case: Circle Although a bit optimised, still O(N^2) technically

QuickHull

- Best-Case: O(n) when hull points are low
- Average-Case: O(N Log(N))
- Worst-Case: Circle O(N^2)

License