**Driver-Assistant Software Instruction**

Jingwen Mo

Yiming Chen

Supervised by Prof. Quoc-Viet Dang

**July 23th, 2018**

# Introduction

## Purpose

The Data-Graph Software is a PC based software developed to draw the graph which depicts different sensors for the racing car. This instruction is for the software users and software tester.

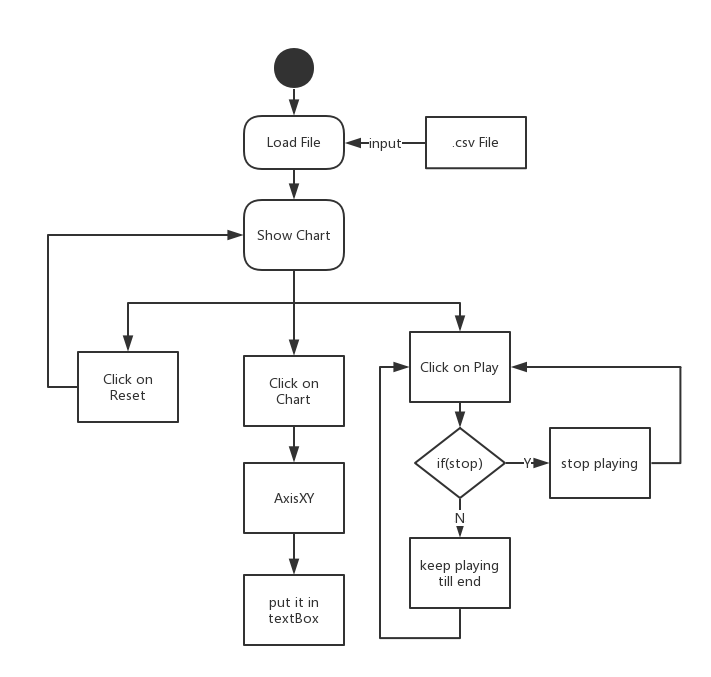
## Background

Racing car is one of the fiercest activities. It requires people to have access to different sensors in the car in order to evaluate the performance of the driver and train the driver by the feedback from the sensors. Our software is meant to help coach to transfer the data of different sensors to intuitive graphs easily.

# Software overview

## Overview

The whole structure of our software could be shown as following flow chart:



## Details

1. Load CSV Files - Click on “Load CSV…” Button

Call function of fileLoadingButton\_Click and pop up a file dialog for users to choose their CSV file. After users choose their files and the columns for latitude and longitude, the software could draw graphs for different series of sensors and the GPS graph for the route of driving. Also, the software would dynamic create textboxes, checkboxes and radio buttons for different sensors.

Related Functions:

void fileLoadingButton\_Click(object sender, EventArgs e)

DataTable OpenCSV(string filePath)

2. Show Specific Value of Points - Click on the chart

Users could click on the graph and show a vertical line at where the mouse click. Then the textboxes below would show the values of different sensors. Meanwhile, there would be a black point in the GPS graph to show the same place as you click on the chart.

Related Functions:

void sensorChart\_MouseClick(object sender, MouseEventArgs e)

int findLeftNear(double value, double[] array, int length)

3. Make the Chart Flow - Click on “Play”, “Stop” and “Reset” button

Users could click on “Play” button and call the function of buttonPlay\_Click. This function enables chartTimer so that it could repeatedly execute the function of chartTimer\_Tick at intervals. In the meantime, the textboxes below would show the values of sensors as the chart flowing. When users click on “Stop” button, the chart would stop flowing. After “Play” button is re-clicked, the chart would begin to flow from where it stops. The “Reset” button would let users to reset the chart to the original state.

Related Functions:

void buttonPlay\_Click(object sender, EventArgs e)

void buttonStop\_Click(object sender, EventArgs e)

void resetButton\_Click(object sender, EventArgs e)

void chartTimer\_Tick(object sender, EventArgs e)

4. Choose Different Type of Y Axis - Click on “R1”, “R2”, “R3” or “R4” radio button

Users could change different series from one Y axis to the other one. The software provides 4 different Y axis. The change of axis could be change by radio buttons dynamically created by fileLoadingButton\_Click function.

Related Functions:

void change(int no, ChartArea caR)

void rb1\_Click(object sender, EventArgs e)

void rb2\_Click(object sender, EventArgs e)

void rb3\_Click(object sender, EventArgs e)

void rb4\_Click(object sender, EventArgs e)

5. Customize X axis and Y axis - Click on “Configuration” button

Users could click on “Configuration” button to pop out a dialog for X axis and Y axis customization. Users could change the range of X axis range, scale and interval and Y axis range and type.

Related Functions and Forms:

public partial class RangeForm : Form

void YRangeForm\_Load(object sender, EventArgs e)

void confirmButton\_Click(object sender, EventArgs e)

void ConfigureButton\_Click(object sender, EventArgs e)

6. Save and Load Setting Log Files - Click on “Saving Setting…” and “Loading Settings” button

By clicking button of “Saving Setting…” in RangeForm, users could save log files with the X axis and Y axis configuration in specific form. Also, users could load log files created by them by clicking the button of “Loading Settings”.

Related Functions:

void settingSaveButton\_Click(object sender, EventArgs e)

void settingButton\_Click(object sender, EventArgs e)

7. Show or Hide Specific Series - Click different checkboxes in displayPanel

User could click on different checkboxes created dynamically by fileLoadingButton\_Click function. By the way, there is an allSelectedCheckBox for the selection of all.

Related Functions:

void sensorCheckedListBox\_ItemCheck(object sender, ItemCheckEventArgs e)

void allSelectedCheckBox\_CheckedChanged(object sender, EventArgs e)

8. Show Colored GPS Graph Changed by Speed or Acceleration - Click different radio buttons in GPSGroupBox

Users could click on radioButton\_Normal to show normal GPS graph, radioButton\_Speed to show GPS graph changed by speed and radioButton\_Accelerate to show GPS graph changed by acceleration.

Related Functions:

void radioButton\_Normal\_CheckedChanged(object sender, EventArgs e)

void radioButton\_Speed\_CheckedChanged(object sender, EventArgs e)

void radioButton\_Accelerate\_CheckedChanged(object sender, EventArgs e)

int colorRed(double x)

int colorGreen(double x)

9. Drag Mouse on the Chart - Just drag the cursor on the chart

Related Functions:

void sensorChart\_MouseMove(object sender, MouseEventArgs e)

void sensorChart\_MouseDown(object sender, MouseEventArgs e)

void sensorChart\_MouseUp(object sender, MouseEventArgs e)

# Testing and Results

## Testing Environment

- Visual Studio 2013&2015

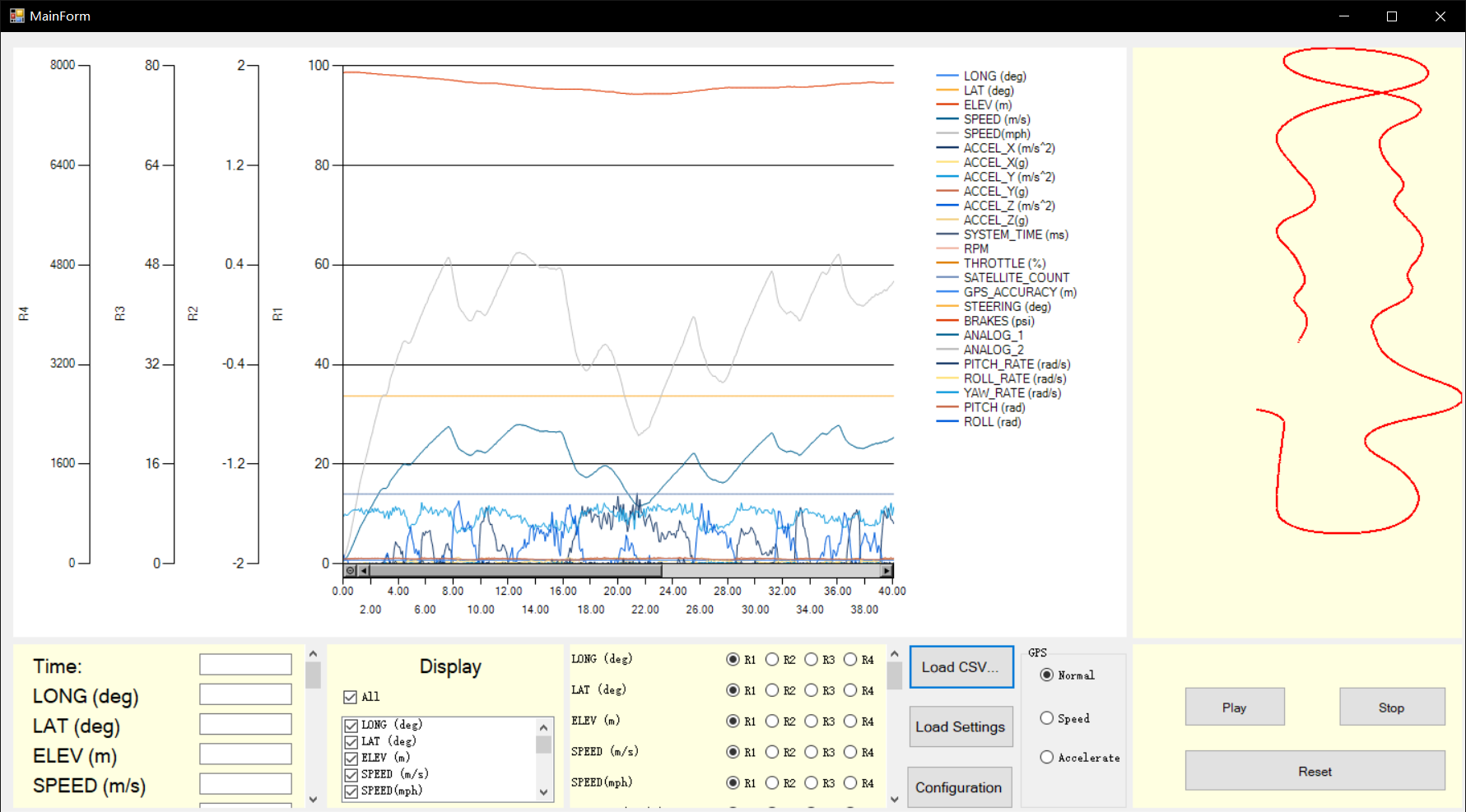
- Programming Language :C#

## Results

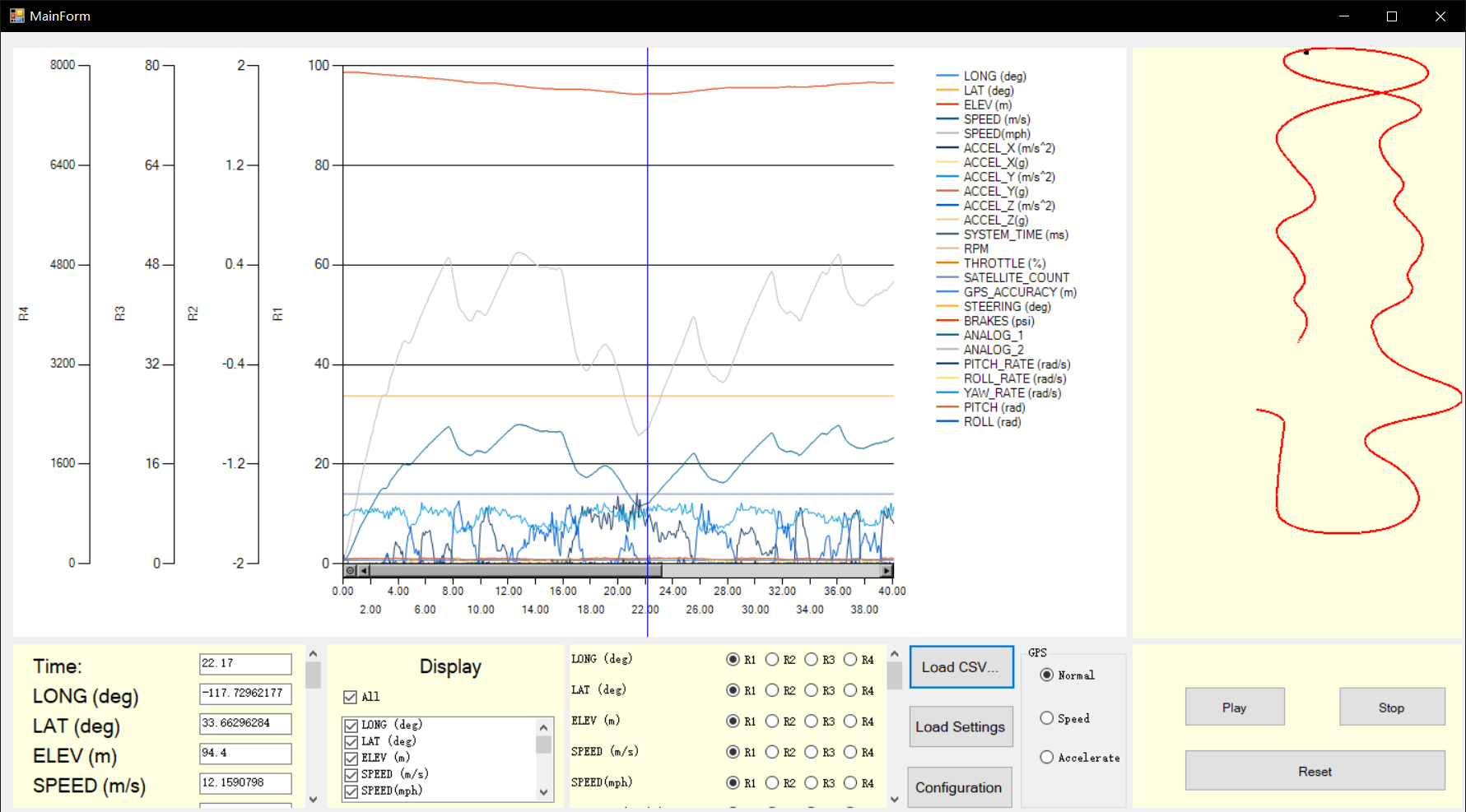
1. Load CSV Files





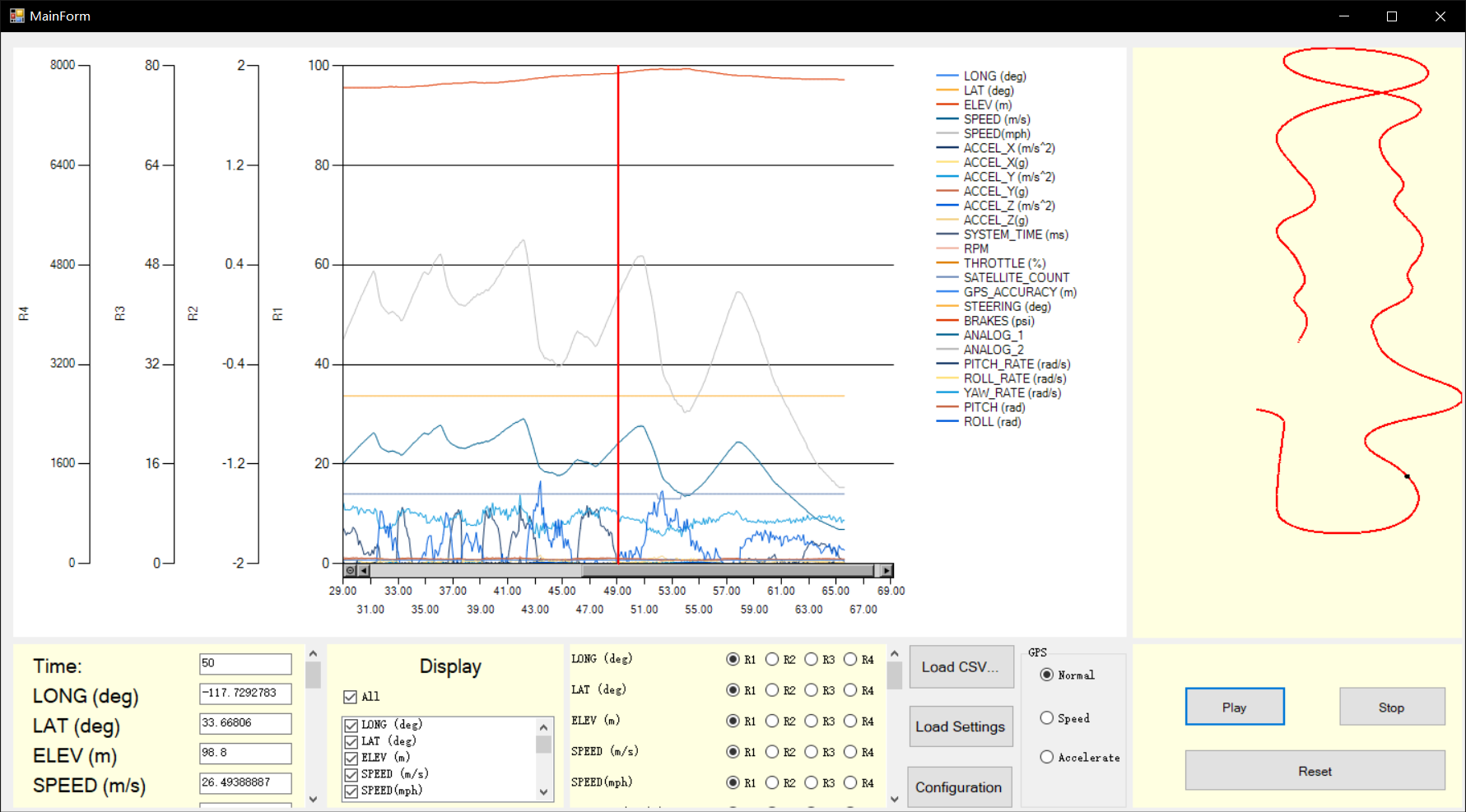


2. Show Specific Value of Points



3. Make the Chart Flow

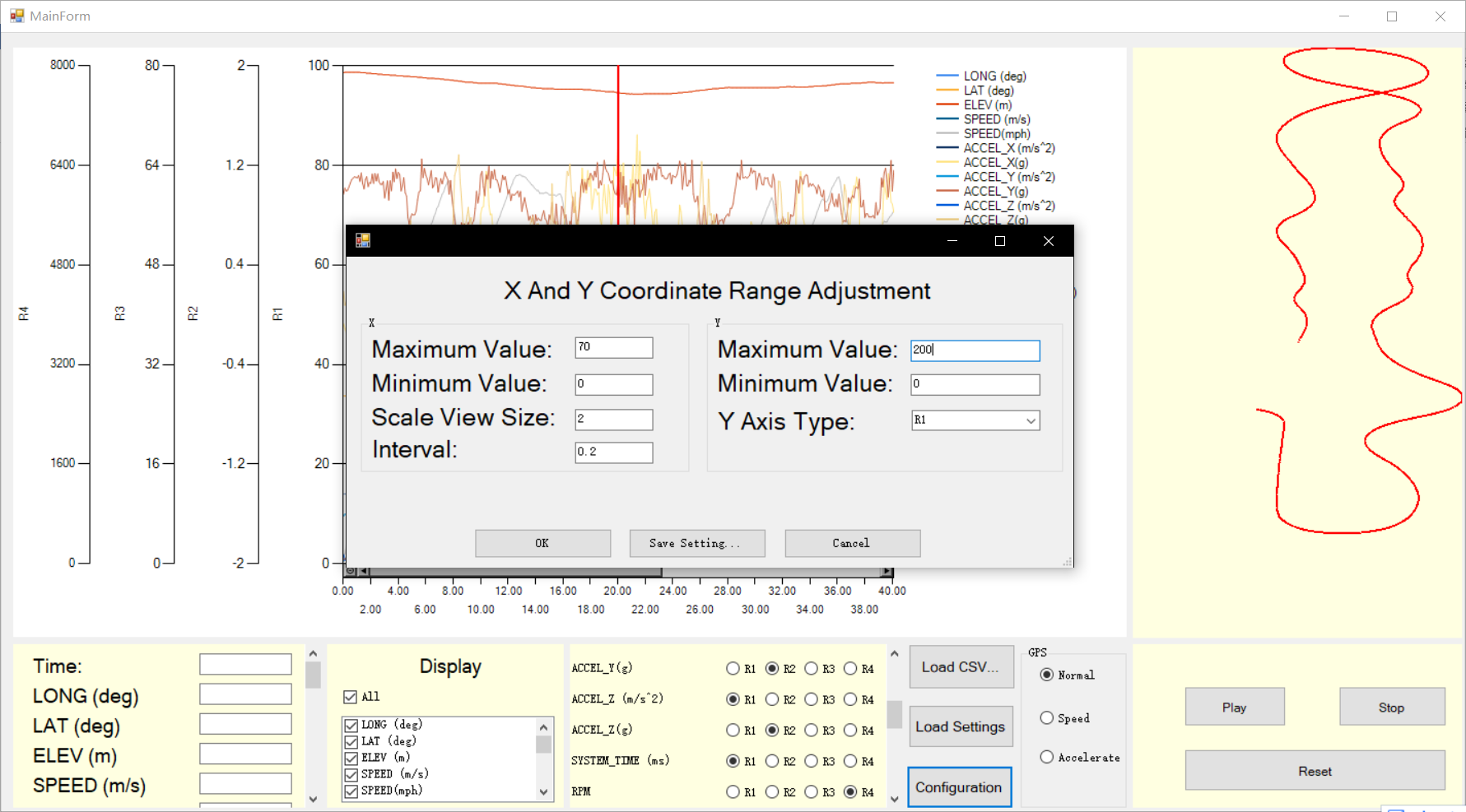




4. Choose Different Type of Y Axis

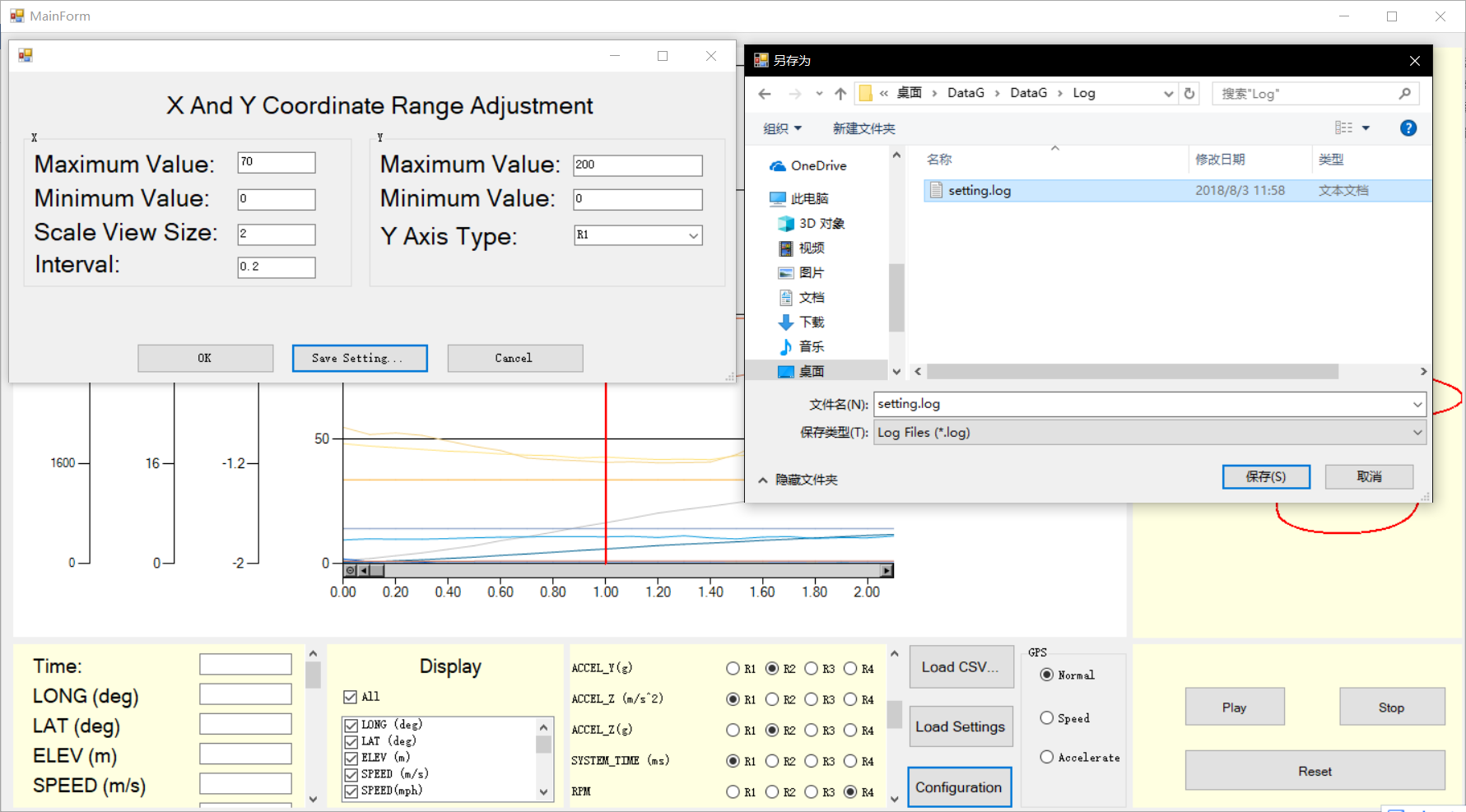


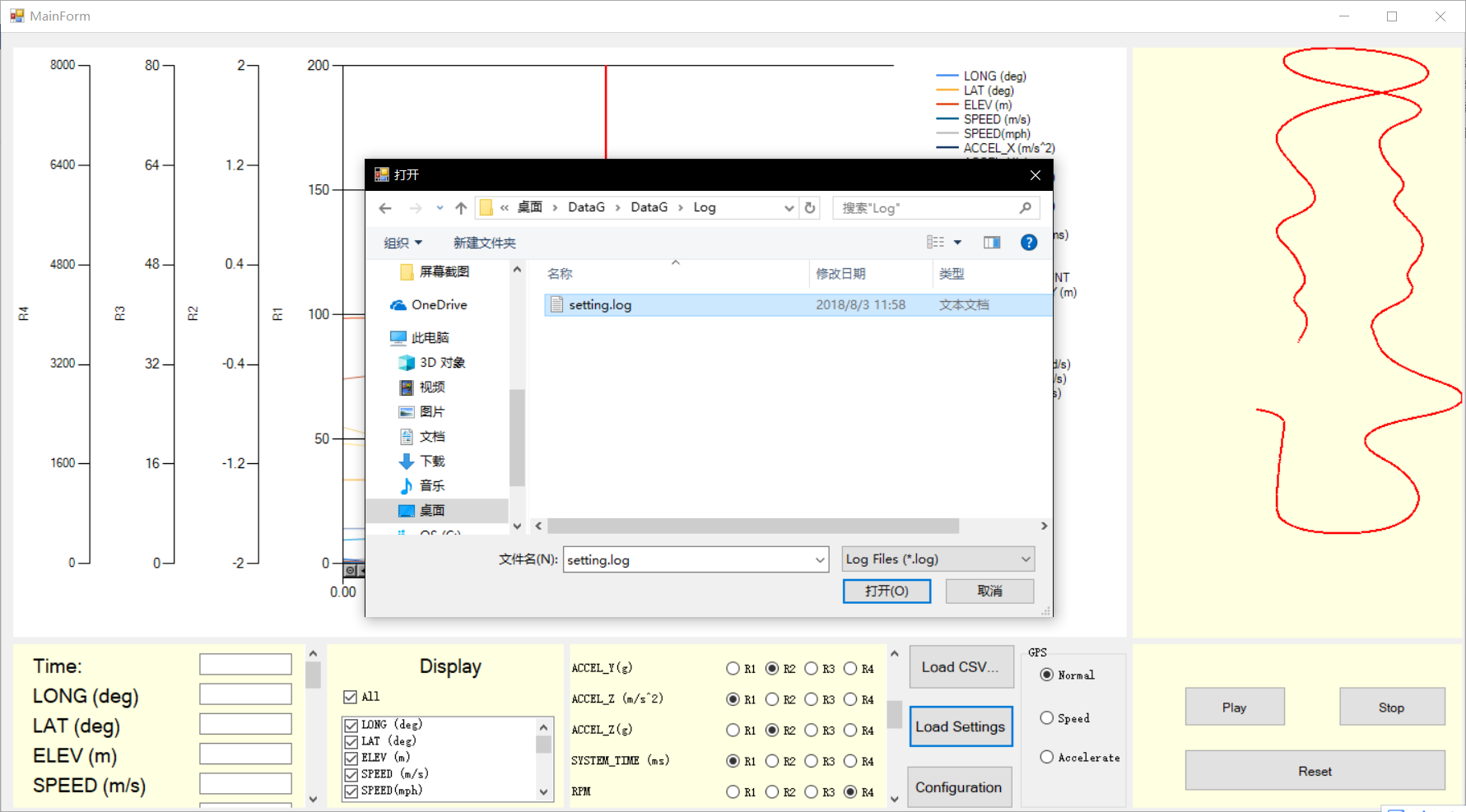
5. Customize X axis and Y axis

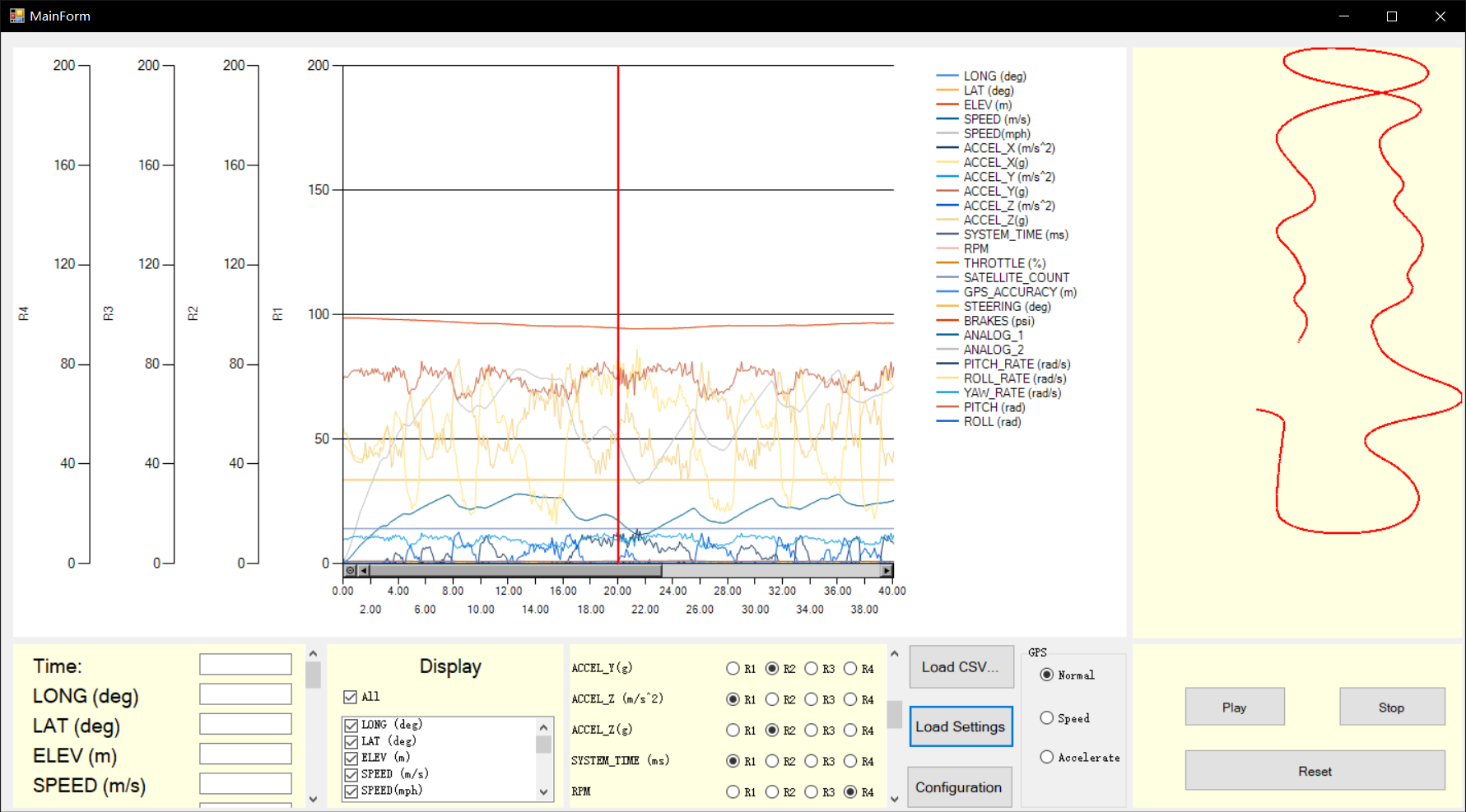




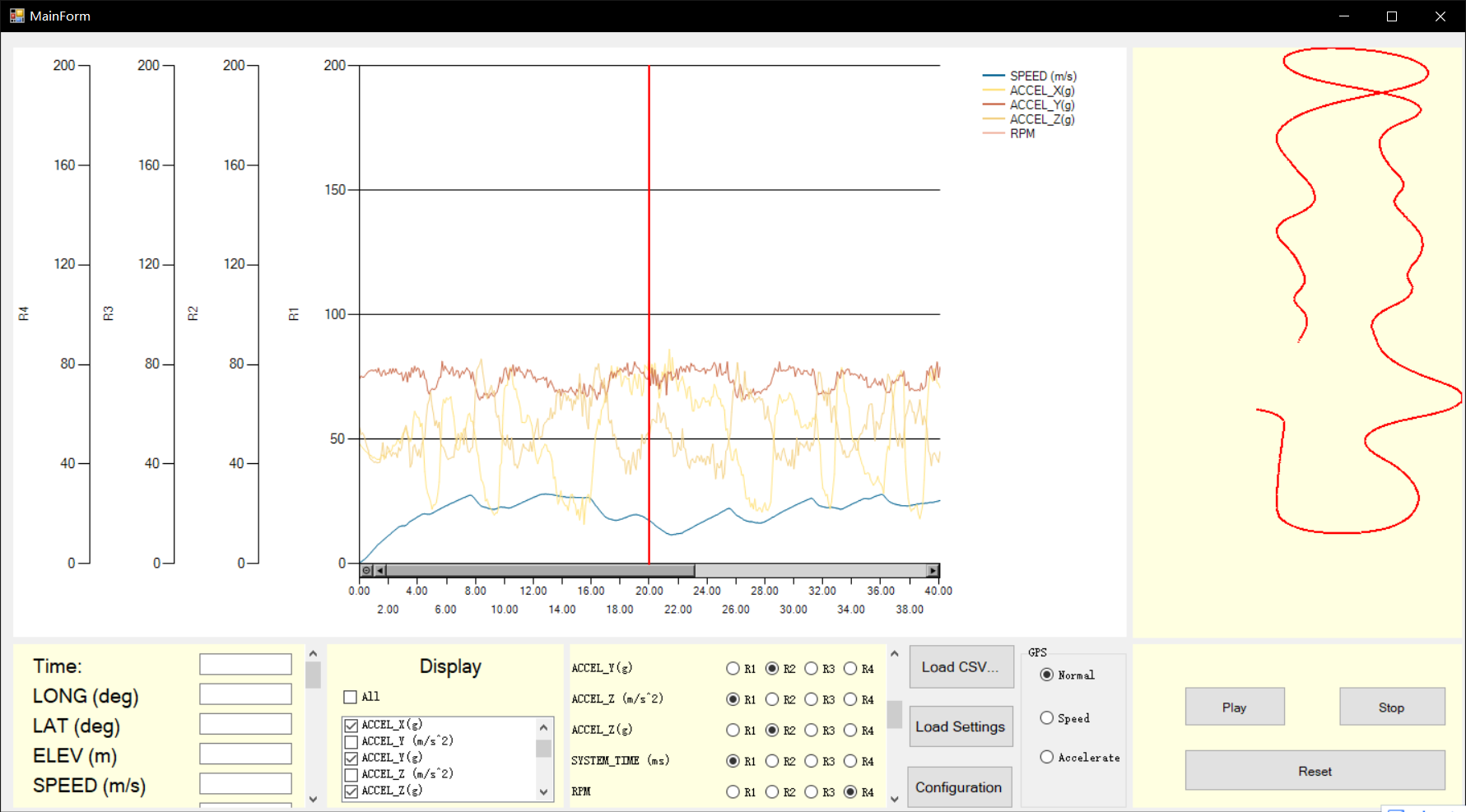
6. Save and Load Setting Log Files



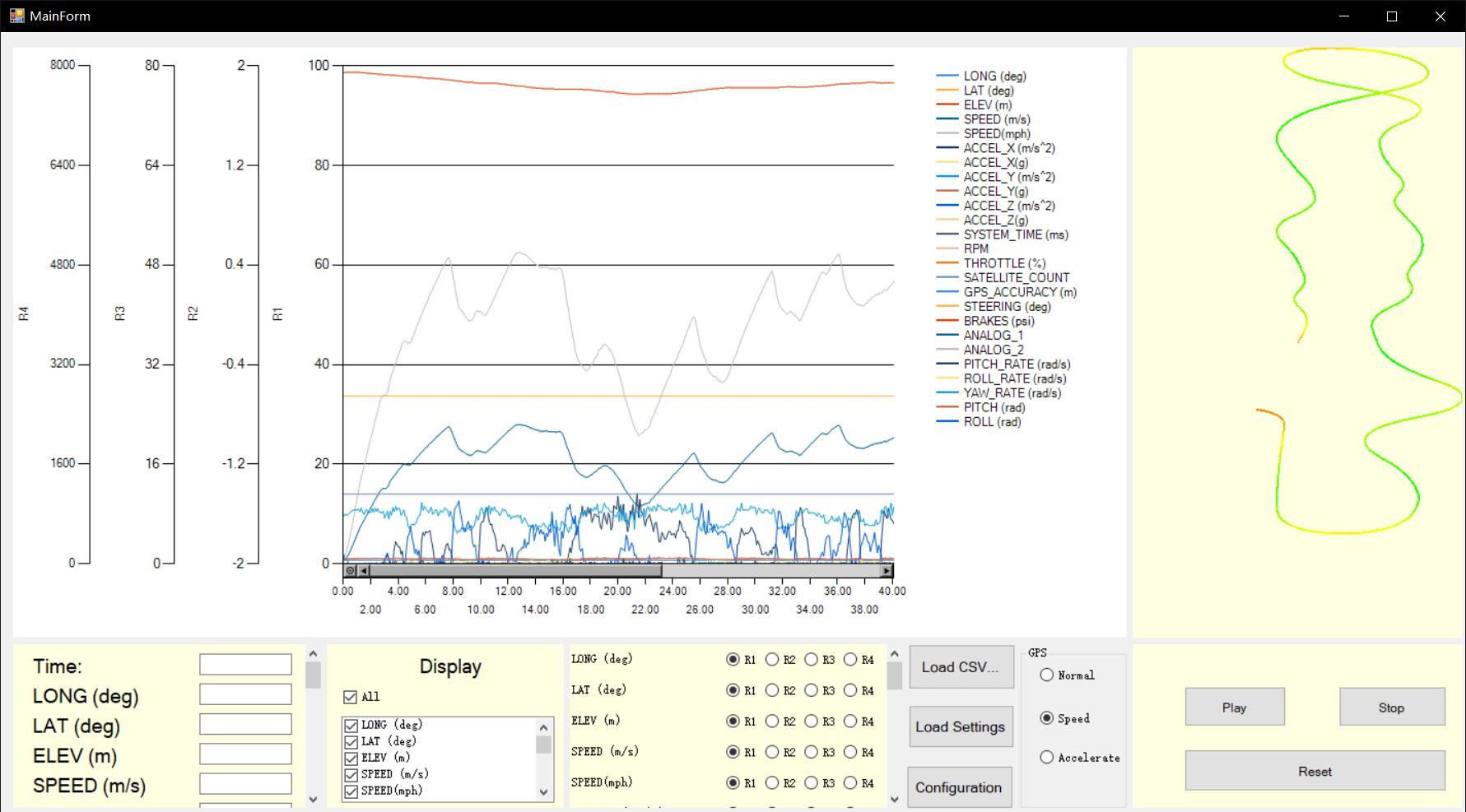




7. Show or Hide Specific Series



8. Show Colored GPS Graph Changed by Speed or Acceleration



9. Drag Mouse on the Chart