

## АННОТАЦИЯ

Программа для ракетного комплекса используется для моделирования полета неуправляемой ракеты типа “Земля-земля”. В приложении учтен разгон по направляющей, изменяемая масса и угол относительно горизонта на активном участке траектории. Неучтенно сопротивление воздуха, шарообразность земли и изменение гравитации от высоты.

Программа является оконным приложением, которое принимает в поля для ввода текста параметры ракеты и условия запуска, выводит траекторию полета ракеты и угол запуска. Выводит конечные результаты, такие как высота подъема, дальность полета.

Программа предназначена для ракет, не выходящих на орбиту. У ракеты типа земля-земля выделяются 3 фазы полета:

1. Разгон по направляющей – ракета начинает свое движение по специальной рельсе, направленной вверх под некоторым углом к горизонту. Такая рельса используется для разгона ракеты. Без нее траектория движения ракеты становится менее предсказуемым;
2. Движение ракеты на активном участке траектории – ракета летит на двигателе, направляя вектор тяги по вектору скорости до тех пор, пока не кончится топливо;
3. Движение ракеты на пассивном участке траектории – ракета летит по инерции, пока не достигнет земли. На этом участке ракета рассматривается как тело, брошенное под углом к горизонту.

## Содержание

ВВЕДЕНИЕ .....	4
1 История создания .....	5
2 Постановка задачи.....	10
3 Анализ предметной области.....	11
4 Используемые методы .....	13
5 Алгоритмы и средства реализации.....	14
6 Перспективы дальнейшей разработки .....	22

## ВВЕДЕНИЕ

Приложение должно было принимать на вход простые данные, которые можно получить в “домашних условиях”, иметь возможность вывода нескольких траекторий для визуального сравнения, иметь вычисляемую и небольшую погрешность вычислений, так как она является накапливаемой.

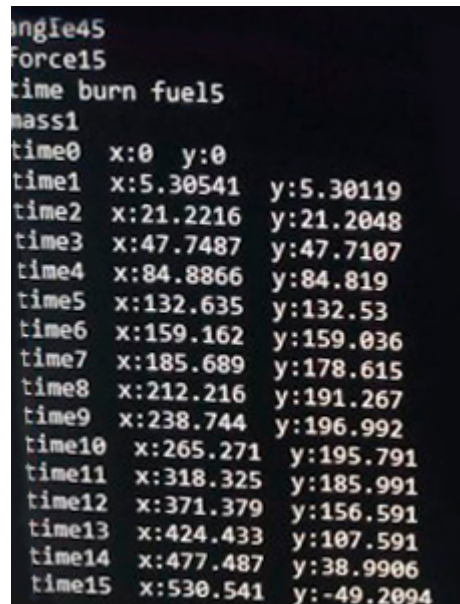
В программе моделируются три участка полета ракеты – разгон по направляющей, активный участок траектории и пассивный участок траектории. Для каждого участка применяются свои методы. Так для разгона по направляющей был написан метод, для которого была выведена формула движения на этом участке. Для активного участка траектории было выведено дифференциальное уравнение, для решения которого был применен метод Рунге-Кутты 4-го порядка. Для пассивного участка траектории применяется закон для тела, брошенного под углом к горизонту.

Благодаря разбиению траектории на три участка и реализации методов в своих классах удалось сделать код программы более читаемым.

По ходу данной работы автор сталкивался и решал следующие проблемы: Разворот ракеты на активном участке, движение ракеты по направляющей, решение дифференциального уравнения, создание оконного приложения.

## 1 История создания

Проект имеет длинную историю создания. Сначала это было консольным приложением C++ (рисунок 1), которое выводило траекторию ракеты в виде координат точек. Масса ракеты была неизменной на всей траектории, не было направляющей, ракета не меняла угол относительно горизонта, из-за чего летала боком.



```
angle45
force15
time burn fuel5
mass1
time0 x:0 y:0
time1 x:5.30541 y:5.30119
time2 x:21.2216 y:21.2048
time3 x:47.7487 y:47.7107
time4 x:84.8866 y:84.819
time5 x:132.635 y:132.53
time6 x:159.162 y:159.036
time7 x:185.689 y:178.615
time8 x:212.216 y:191.267
time9 x:238.744 y:196.992
time10 x:265.271 y:195.791
time11 x:318.325 y:185.991
time12 x:371.379 y:156.591
time13 x:424.433 y:107.591
time14 x:477.487 y:38.9906
time15 x:530.541 y:-49.2094
```

Рисунок 1 – Консольное приложение

Далее возникла идея визуализировать эту траекторию, однако у автора не было времени осваивать win-api. В результате поиска конструктора для создания оконного приложения, такой конструктор обнаружился в шаблоне проекта C#, после чего проект преобразился в оконное приложение с удобным вводом и визуализацией траектории. Полем для графика являлся “холст”. Траектория, оси и масштабная сетка рисовались программой с помощью кисти (рисунок 2).

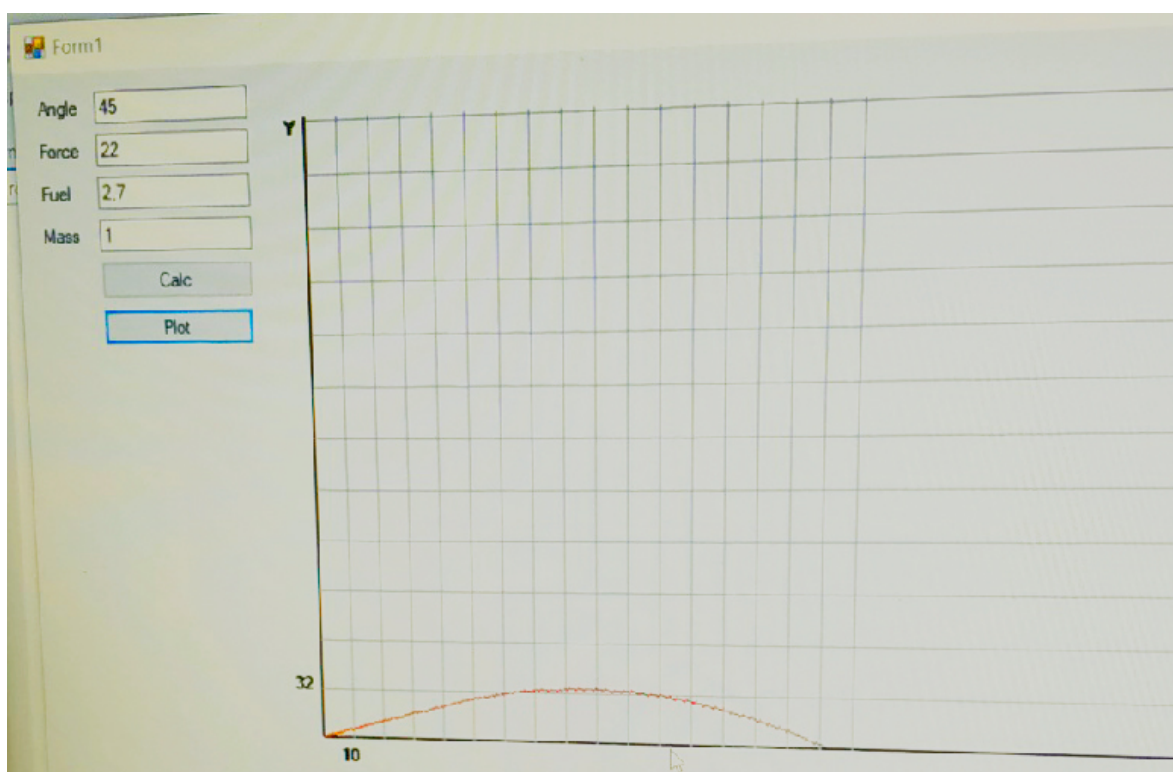


Рисунок 2 – Оконное приложение C#

Так как ракета не поворачивалась во время полета, то есть летала боком, было принято решение поворачивать ракету во время полета по вектору скорости. Для разворота во время полета было составлено дифференциальное уравнение, которое решалось методом прямоугольников. Тогда же после изучения видеоматериалов и литературы по любительскому ракетостроению в программу была добавлена направляющая для разгона (рисунок 3).

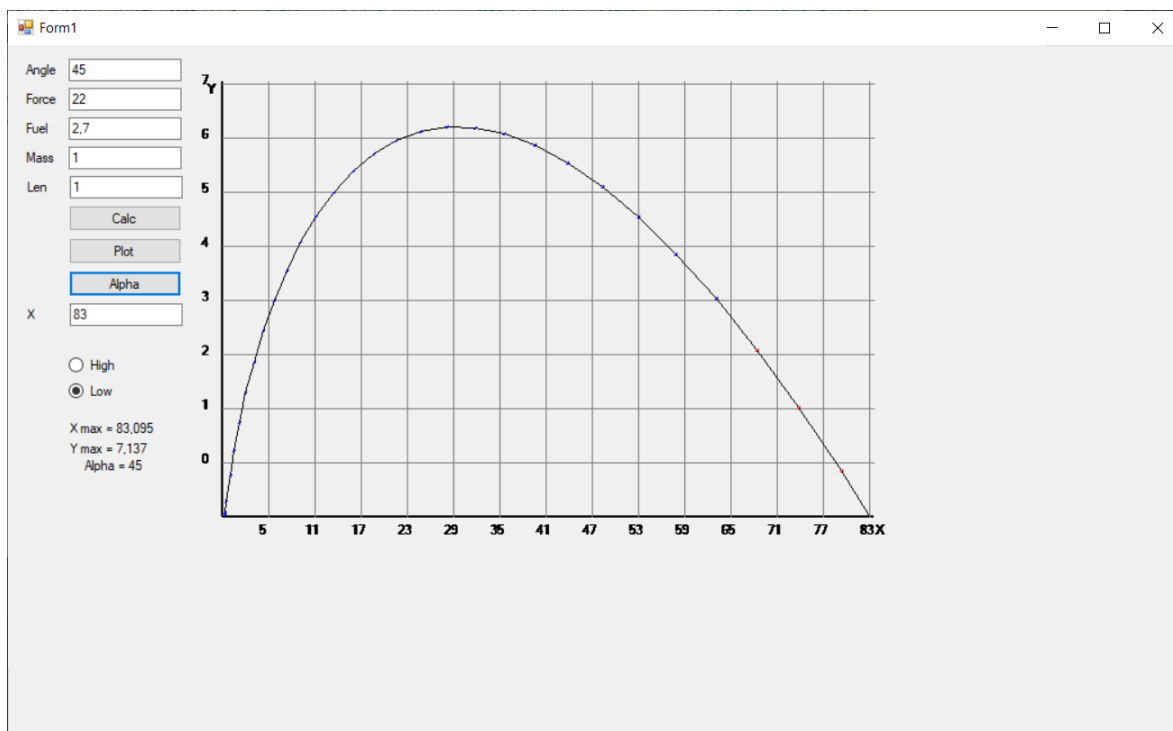


Рисунок 3 – Визуализация новой траектории ракеты

Язык C# был новым для автора, поэтому процесс разработки был долгим, так как многое приходилось искать в интернете. После тщательного поиска, было обнаружено, что у языка C++ то же есть конструктор оконных приложений, однако его надо было подключать отдельно (он не входил в шаблон). Так проект был перенесен на C++.

Во время изучения теории по созданию оконных приложений был обнаружен объект Chart, который принимает массив точек, а выводит график, состоящий из них, сам рисует оси и масштабную сетку.

Так же в программу была добавлена возможность посчитать угол, при котором ракета преодолееет заданное пользователем расстояние.

После анализа траекторий, полученных в результате работы программы встала задача вставить в программу изменяемую массу ракеты. было принято решение – каждый раз, когда программа считает новую точку, она вычитала из массы топлива, оставшуюся на предыдущем шаге вычитала массу израсходованного топлива, которое сгорело за промежуток времени, однако

такой метод имеет погрешность, и через некоторое время было составлено уравнение движения по направляющей, в которой учитывается расход топлива.

После тестов программы, оказалось, что вычисления имеют огромную накапливаемую погрешность, которая получалась из-за численного решения дифференциального уравнения методом прямоугольников. Это было обнаружено, когда во время программного эксперимента был изменен шаг вычисления новой точки.

Следующим шагом стало добавление в проект его символа (рисунок 4).

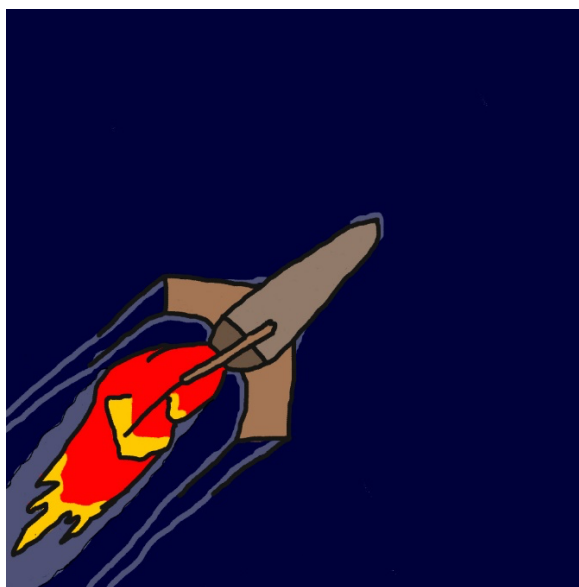


Рисунок 4 - Иконка

Для анализа разных траекторий была добавлена возможность рисовать несколько траекторий на одном поле (рисунок 5).

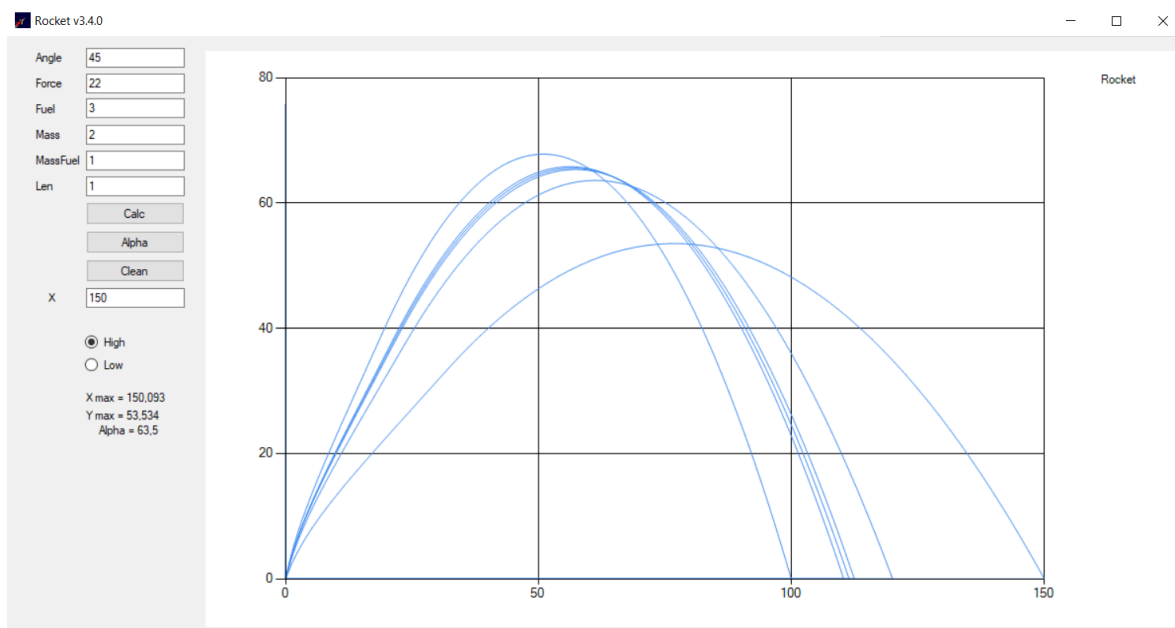


Рисунок 5- Визуализация нескольких траекторий

Было необходимо найти более точные методы для решения уравнения. Так был найден метод Рунге-Кутты 4-го порядка. Не удалось разобраться, как он работает, однако получилось вставить его в программу. С новым методом удалось посчитать погрешность и вывести ее график (рисунок 6).

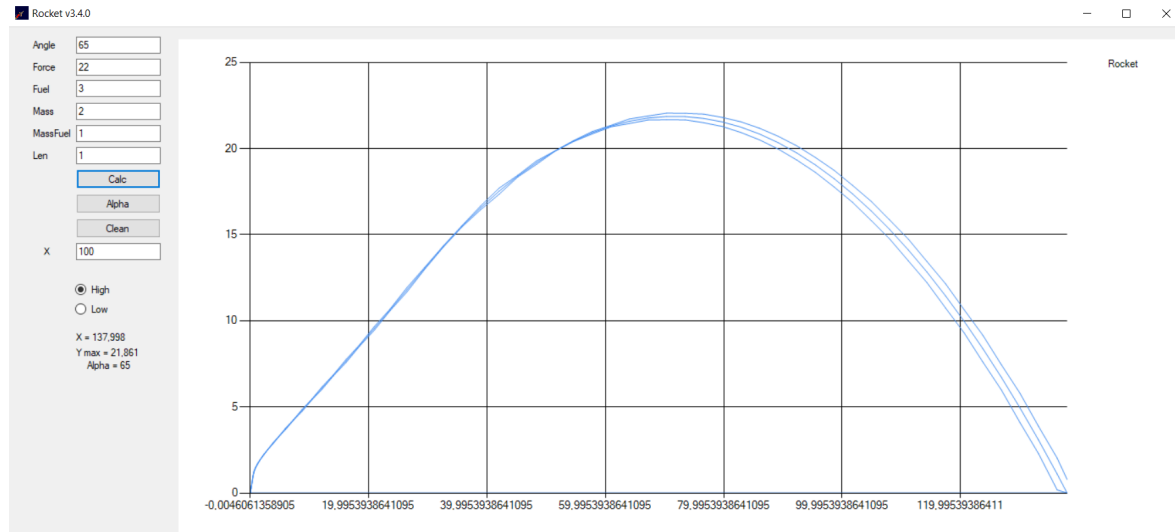


Рисунок 6 – Туннель погрешности



## 2 Постановка задачи

Было необходимо создать приложение для расчета и визуализации траектории ракеты, расчета угла запуска и нахождения расстояния до точки приземления и иметь незначительную погрешность, которое должно работать в двух режимах:

- рассчитывать и выводить на экран траекторию полета ракеты при заданном угле.
- Рассчитывать и выводить на экран траекторию полета ракеты при заданном расстоянии.

Пользователь вводит:

- Силу тяги двигателя.
- Вес заправленной ракеты.
- Массу топлива.
- Длину направляющей.
- Время горения топлива.

Далее, если искомым является угол, то пользователь вводит дальность полета.

В том случае, если искомым является дальность полета, пользователь вводит угол взлета.

В программе должна быть возможность вывода нескольких траекторий в поле для графика.

### 3 Анализ предметной области

Очевидно, что полет ракеты состоит из трех последовательных фаз:

Для расчета движения по направляющей была составлена формула (1), где путь является первообразной от скорости.

$$s = \int v \times dt = \frac{F}{\mu^2(\mu \times t - m_0)} \times \ln\left(\frac{m_0}{m_0 + \mu \times t}\right) + F \times \frac{t}{\mu} - g \times \sin(\alpha) \times \frac{t^2}{2}, \quad (1)$$

где:  $s$  – путь [м].

$v$  – скорость [м/с<sup>2</sup>].

$t$  – время [с].

$F$  – тяга двигателя [Н].

$\mu$  – расход топлива [кг/с].

$m_0$  – масса заправленной ракеты [кг].

$g$  – ускорение свободного падения [м/с<sup>2</sup>].

Для расчета траектории ракеты на активном участке была выведена система дифференциальных уравнений (2) и (3).

$$V^0x = \frac{VxF}{M\sqrt{V^2x+V^2y}} \quad (2)$$

$$V^0y = \frac{VyF}{M\sqrt{V^2x+V^2y}} - g \quad (3)$$

где  $V^0x$  – производная по времени от скорости по оси  $x$ , м/с<sup>2</sup>.

$M$  – масса ракеты на данный момент, кг.

$V^0y$  – производная по времени от скорости по оси  $y$ , м/с<sup>2</sup>.

$F$  – Тяга двигателя, Н.

$g$  – ускорение свободного падения, м/с<sup>2</sup>.

Для расчета траектории ракеты на пассивном участке была применена формула для тела, брошенного под углом к горизонту.

$$x = v * \cos(\alpha) * t$$

$$y = v * \sin(\alpha) * t - (g * t^2)/2$$

где  $\alpha$  – угол, радианы.

$t$  – время, с.

$g$  – ускорение свободного падения,  $\text{м/с}^2$ .

$v$  – скорость,  $\text{м/с}$ .

$x$  – координата по оси  $X$ .

$y$  – координата по оси  $Y$ .

#### 4 Используемые методы

Для разработки приложения использовалась:

- Язык программирования C++
- Microsoft Visual Studio
- Стандартная библиотека шаблонов STL(C++)
- Метод Рунге-Кутты 4-го порядка для расчета траектории ракеты на активном участке.
- Правило Рунге для расчета погрешности от шага.

## 5 Алгоритмы и средства реализации

В программе не учитывается сила сопротивления воздуха, а ракета представлена как материальная точка с вектором тяги и массой. Для того чтобы траектория ракеты была схожа с реальной, было принято решение разворачивать ракету по направлению вектора скорости.

Временной промежуток между соседними точками составляет 0.1 секунду. Точки записываются в массив типа `vector`

Пассивная фаза полета длится пока ракета не достигнет земли. Так как новая точка ставится каждую 0.1 секунду, то по оси  $Y$  ракета уходит в минус, для того чтобы найти последнюю точку программа берет предпоследнюю и рассчитывает точку приземления исходя из высоты, скорости и ускорения ракеты.

Метод золотого сечения – используется в программе для нахождения максимальной дальности полета ракеты (рисунок 7).

На первой итерации заданный отрезок делится двумя симметричными относительно центра точками и рассчитываются значения для каждой из точек, после чего конец отрезка, ближе к которому оказалась та из вновь поставленных точек, значение которой больше переносится в эту точку. На следующей итерации, благодаря свойству метода золотого сечения надо искать всего одну новую точку. Эти действия повторяются, пока расстояние между двумя точками не станет меньше или равно поставленной точности. Далее, для увеличения точности, координаты двух точек складываются и делятся на 2. Полученная координата и будет точкой максимума.

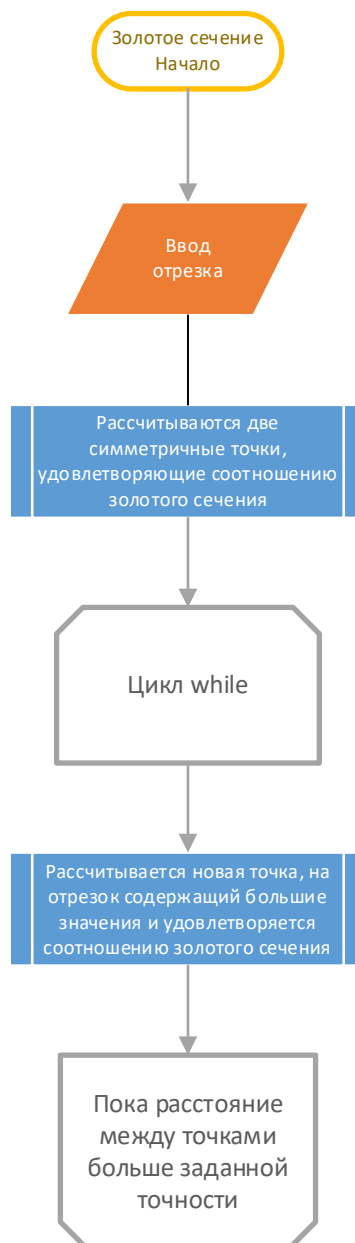


Рисунок 7 – Метод золотого сечения

Метод деления пополам – используется в программе для нахождения угла, при котором ракета преодолеет заданное пользователем расстояние. Программа вычисляет середину заданного отрезка, далее рассчитав интервал значений на двух отрезках, программа определяет на какой из двух частей лежит решение. Отрезок, в интервал которого не входит решение отбрасывается, а оставшийся делится пополам и происходят те же самые действия, пока длина отрезка не будет меньше или равно поставленной точности. Далее для увеличения точности координаты начала отрезка складываются с координатами

конца и делятся на 2. Полученная координата и есть угол запуска ракеты, при котором ракета улетит на заданное расстояние.

Для расчета погрешности от шага в методе Рунге-Кутта было применено правило Рунге. Метод работает так: траектория ракеты считается на двух масштабных сетках, один раз с шагом  $h$ , другой раз с шагом  $h/2$ . Далее из координаты, полученной на первой сетке, вычитается координата полученная на второй, результат берется по модулю и делится на 15 (для метода Рунге-Кутта 4-го порядка). Таким образом получается график погрешности по оси  $Y$  по промежутку времени. Для наглядной визуализации данной погрешности было принято решение из координат траектории ракеты вычесть координаты траектории погрешности для получения графика нижней границы погрешности, а для нахождения верхней – складывать координаты траектории ракеты с координатами траектории погрешности. В результате проведенных действий получился “туннель” погрешности. Линия между двумя границами туннеля – траектория ракеты (рисунок 8).

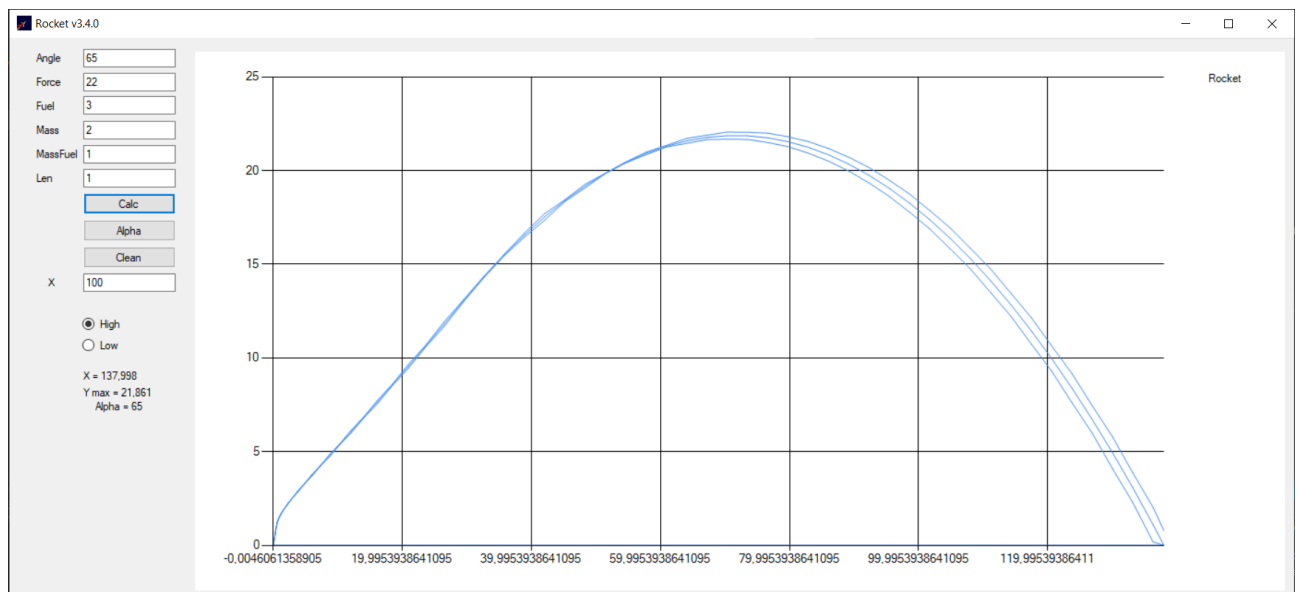


Рисунок 8 – Туннель погрешности

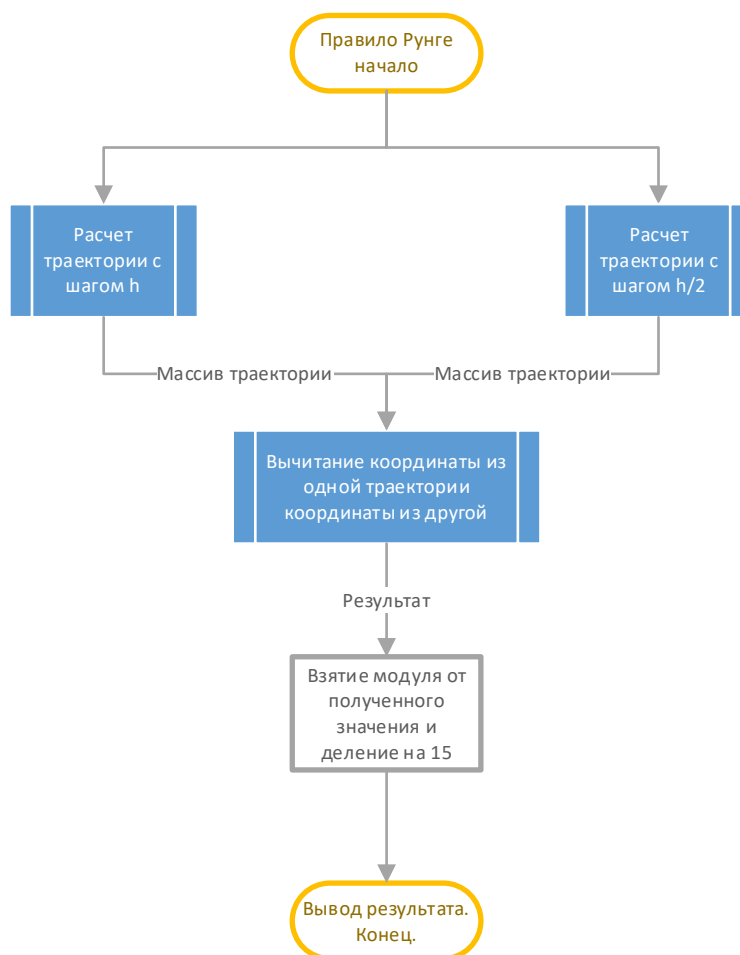


Рисунок 9 - Блок-схема правила Рунге

Класс Phase содержит в себе функции перегрузки операторов, для арифметических операций с объектами. Используется для реализации метода Рунге Кутты (рисунок 9). Объект класса phase имеет скорость по оси X и Y и координаты по оси Y и X.



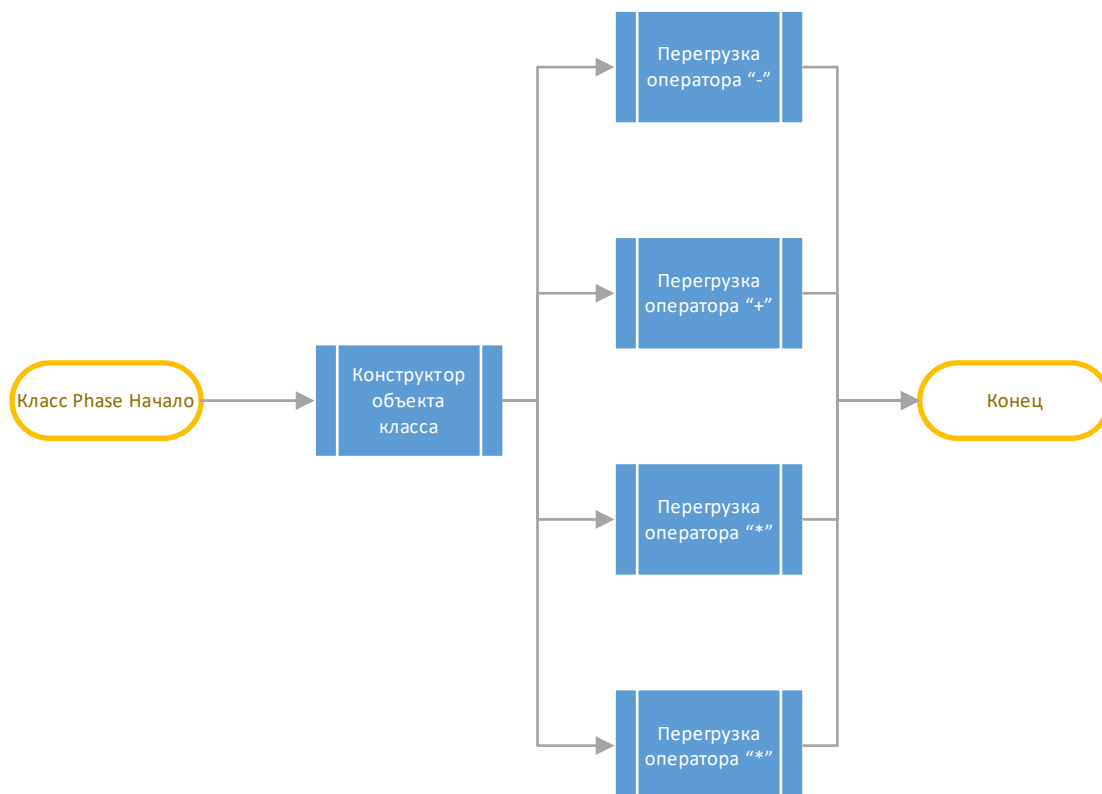


Рисунок 10 - Блок-схема класса Phase

К классу Model (рисунок 11) подключен класс Phase. Класс Model содержит в себе функции для расчета траектории ракеты по направляющей и свободное падение. Для математических функций подключена библиотека math.h.

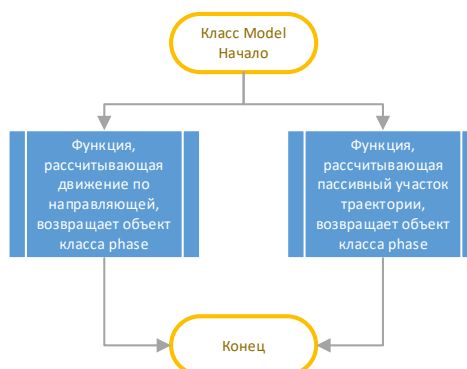


Рисунок 11- Блок-схема класса Model

К классу Numeric (рисунок 12) подключен класс Model. Содержит в себе расчет коэффициентов и реализацию метода Рунге-Кутты. Для расчета используются объекты и функции, описанные в других классах.

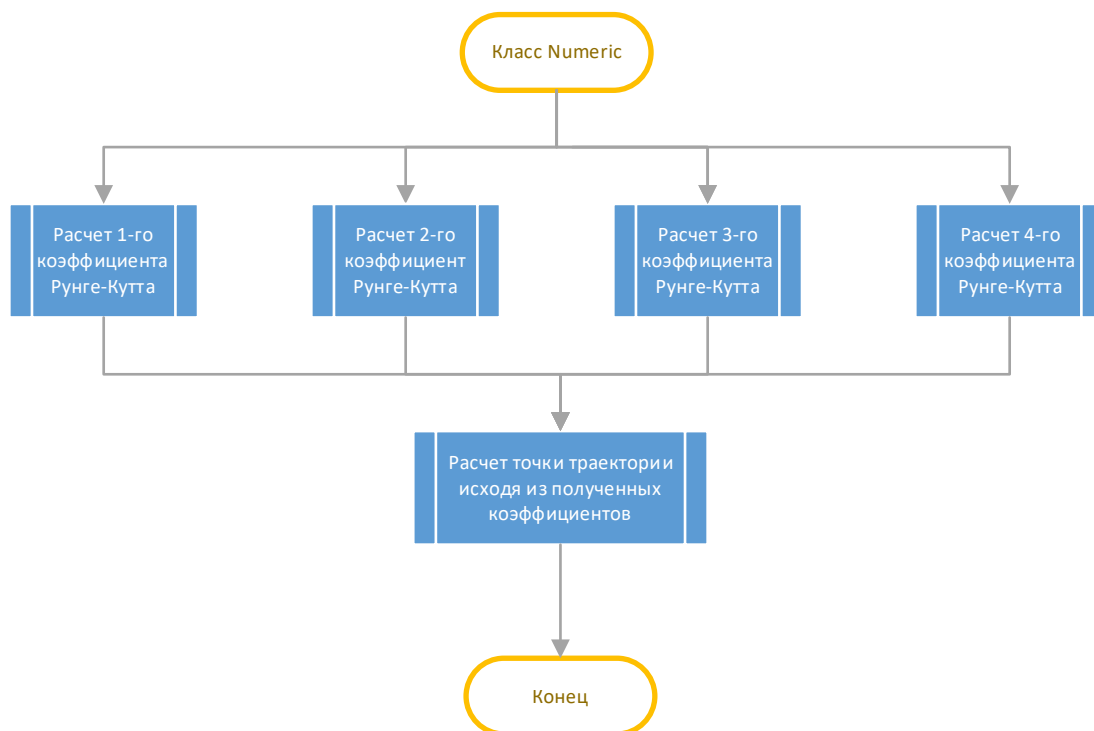


Рисунок 12 – Блок-схема класса Numeric

К классу Counting (рисунок 13) подключен класс numeric, подключаются библиотеки string и vector. В этом классе происходит расчет всей траектории ракеты. Для расчета используются методы, описанные в вышеупомянутых классах, в которые передаются значения для расчета, полученные из текстовых полей на форме.

:

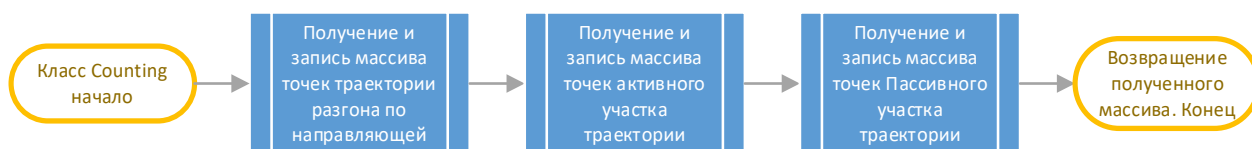


Рисунок 13 - Блок-схема класса Counting

Файл MyForm (рисунок 14) является точкой входа в программу. Здесь элементы (кнопки, текст-боксы, лейблы и так далее) наносятся на форму (окно программы), отрабатываются нажатия на кнопку Calc и Alpha.

Если пользователь нажал на кнопку Calc, класс Counting принимает значения из текст-боксов и рассчитывает траекторию полета ракеты, возвращает полученный массив, который принимает объект chart и выводит визуализированную траекторию. Далее программа выводит угол запуска (значение из текст-бокса Angle) максимальной высоты полета, и дальности (Из массива, который возвращает метод класса counting).

Если пользователь нажал на кнопку Alpha, программа запускает функцию золотого сечения, которая находит угол, при котором ракета улетит на наибольшее расстояние, далее запускает проверку, в которой сравнивает дальность, введенную пользователем с максимальной, в случае если дальность, введенная пользователем, превышает максимальную, программа выводит сообщение об ошибке, в котором содержится максимальная дальность.

Для нахождения угла, при котором ракета улетит на заданное пользователем расстояние, программа использует для расчета интервал углов от 0 до 90 так как для полета на дистанцию меньше максимальной ракета имеет два угла запуска, то, чтобы разграничить эти углы в программу был добавлен переключатель high/low.

Если переключатель стоит в положении high программа рассматривает интервал от угла при полете на наибольшее расстояние до 90. Далее программа запускает метод деления пополам, с помощью этого метода находит угол для полета на заданное расстояние.

Если переключатель стоит в положении Low, то программа рассматривает интервал от 0 до угла при полете на наибольшее расстояние и выполняет действия, описанные выше.

Далее программа выводит угол запуска (полученный в результате работы метода деления пополам) максимальной высоты полета, и дальности (Из массива, который возвращает метод класса counting).

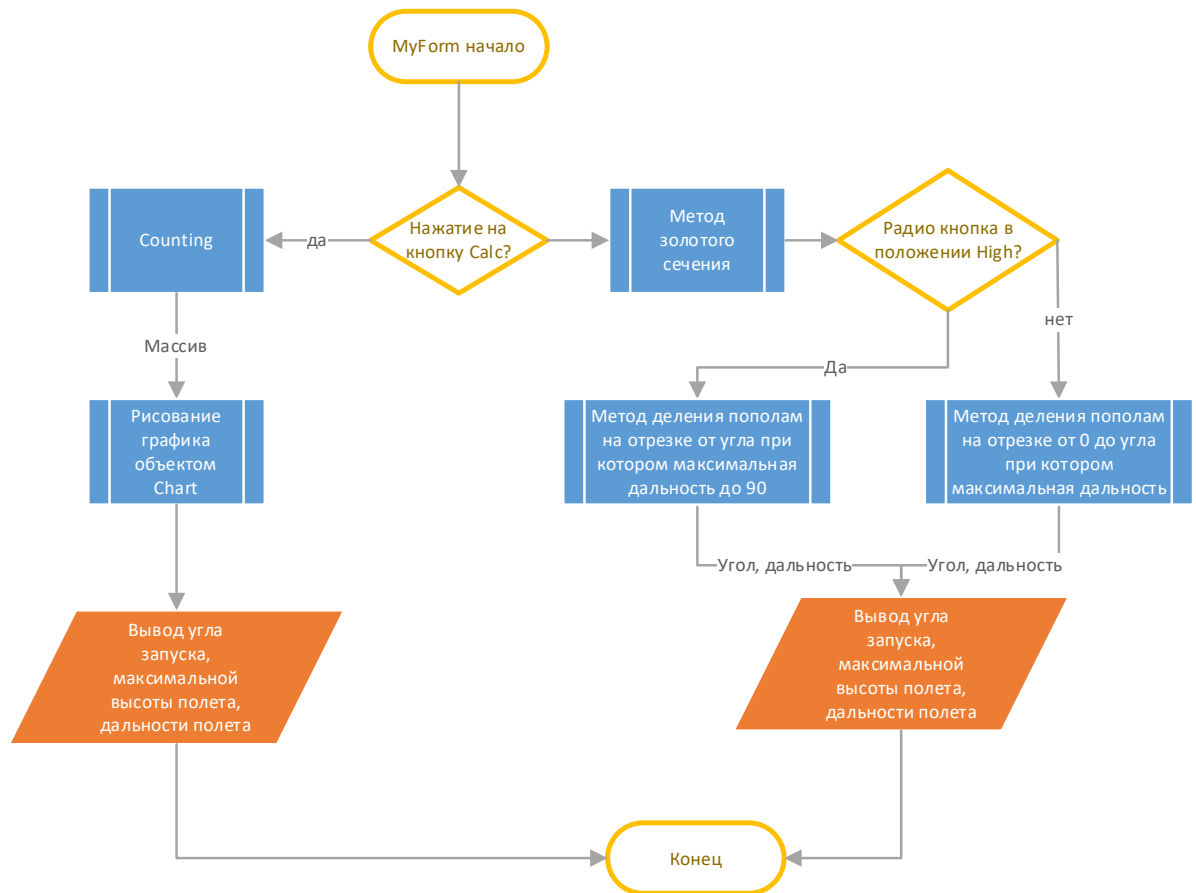


Рисунок 14 - Блок-схема MyForm

## 6 Перспективы дальнейшей разработки

В будущем планируется добавить в программу силу сопротивления воздуха, изменение гравитации в зависимости от высоты, это сделает траекторию, полученную с помощью программы сильно приближенной к реальной.

Учитывать шарообразность земли необходимо при полете на длинные дистанции. Это поможет уменьшить погрешность.

Добавить возможность моделировать полет для многоступенчатых ракет, это увеличит круг моделируемых программой объектов.

Рассматривать ракету как тело с геометрическими характеристиками необходимо, чтобы добавить в программу сопротивление воздуха, которое из-за больших скоростей сильно влияет на траекторию ракеты.

Добавить базу материалов, в которые можно добавлять свои, что бы программа выводила пользователю возможные материалы для строительства ракеты. Это введение должно сильно ускорить процесс разработки ракеты.

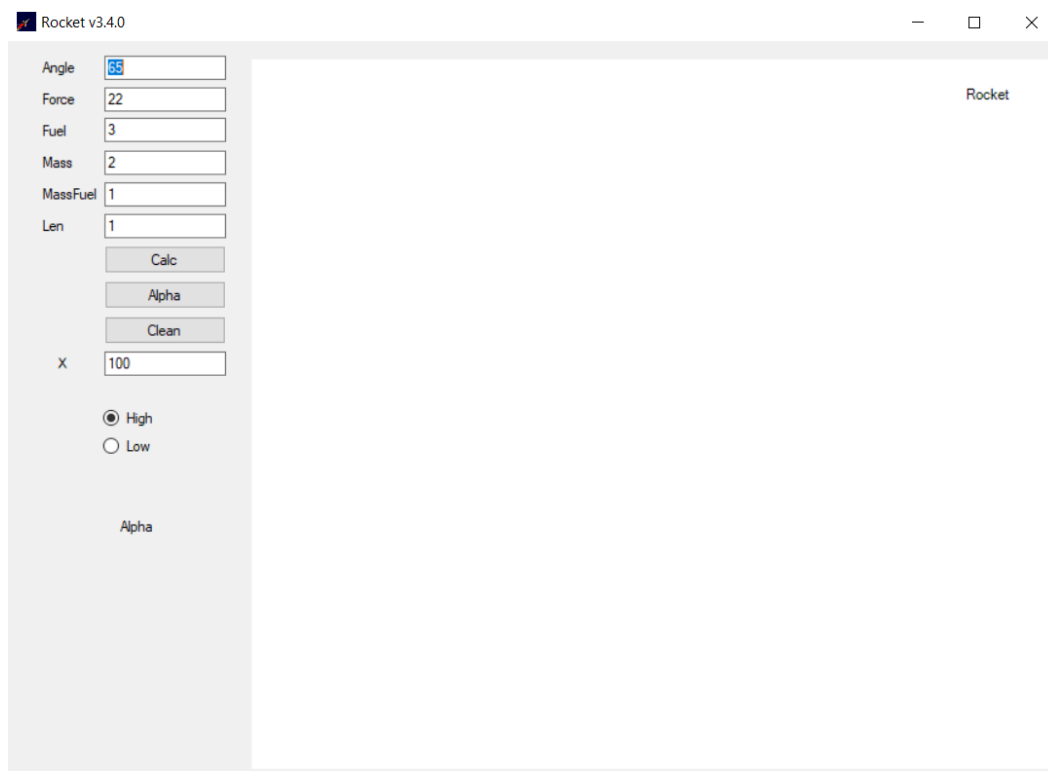


Рисунок 17 – Интерфейс программы (после запуска)

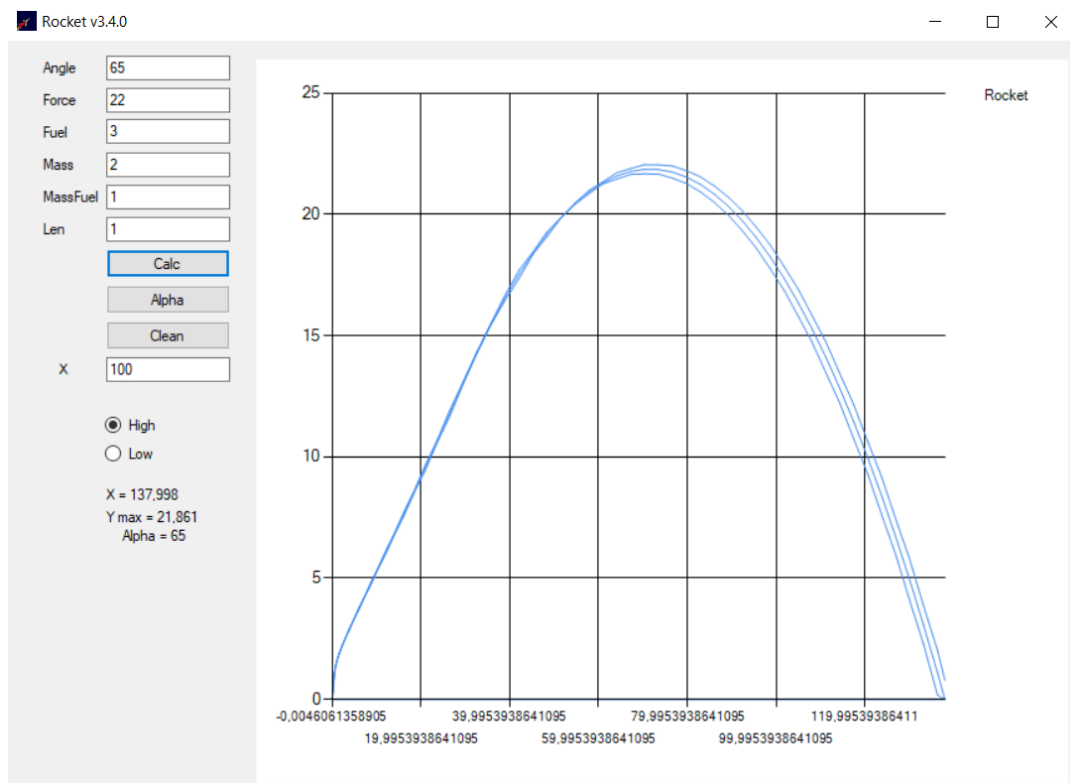


Рисунок 18 – Интерфейс программы (после ввода данных и нажатия на кнопку Calc)

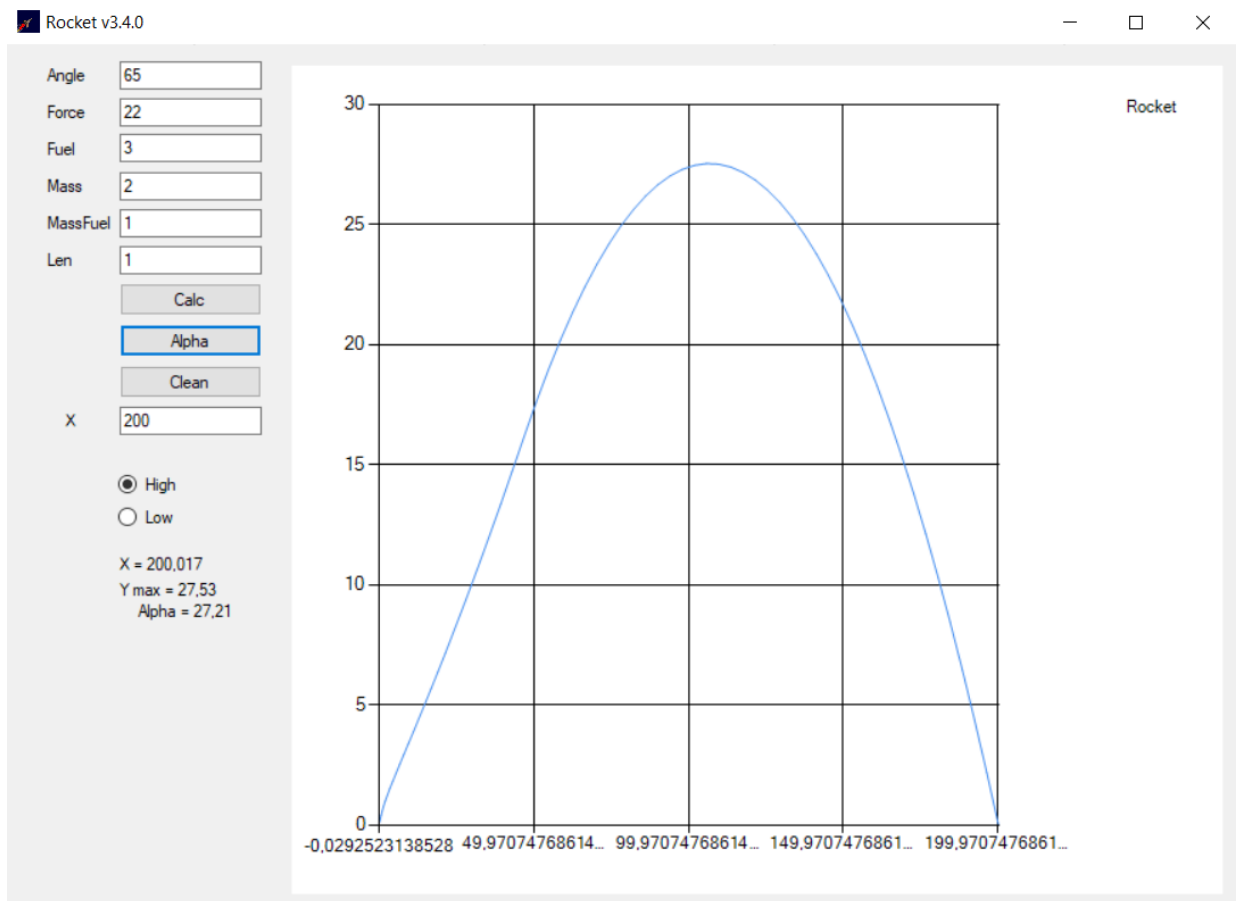


Рисунок 19 – Интерфейс программы (после нажатия на кнопку Alpha)

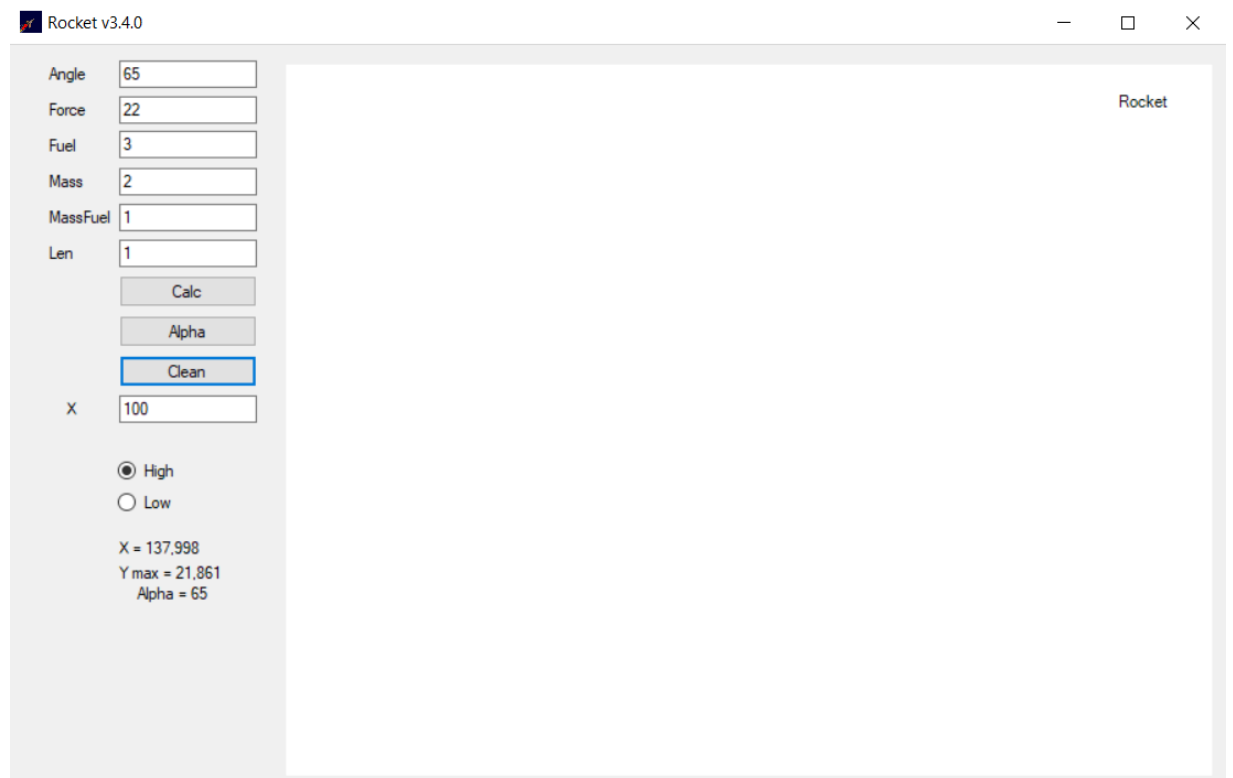


Рисунок 20 – Интерфейс программы (после нажатия на кнопку Clean)