Муниципальное бюджетное общеобразовательное учреждение

Городского округа Балашиха

«Гимназия №1 имени Героя Российской Федерации А.В. Баландина»

МУНИЦИПАЛЬНЫЙ КОНКУРС
ИССЛЕДОВАТЕЛЬСКИХ И ПРОЕКТНЫХ РАБОТ УЧАЩИХСЯ
«ПОТЕНЦИАЛ – 2022»

Практико-ориентированный проект

«Разработка виртуальной лабораторной работы «Бросок под углом»»

Секция № 5: «Что не открыл Архимед, откроем мы!»

Автор работы:

Жокин Александр,

11 «Г» класс

Руководитель:

Куринова Марина Анатольевна,

учитель физики МБОУ «Гимназия №1»

2022 г.

Оглавление

1. Введение 3
2.Основная часть. 4
2.1.Исследование интернет - ресурсов. 4
2.2. Godot Engine. 6
2.3. Моделирование. 6
2.3.1. Движение тела под действием силы тяжести. 7
2.3.2. Движение заряженной частицы под действием электрического поля 8
2.4. Разработка виртуальной лабораторной работы. 9
2.4.1. Разработка программной модели броска тела. 9
2.4.2. Разработка программной модели полёта частицы. 10
2.5. Разработка образцов лабораторных работ и исследовательских задач. 10
3. Заключение. 12
4. Список источников и литературы. 12

1. Введение

Подготовка к государственной итоговой аттестации волнует многих учащихся выпускных классов. По физике во многих разделах применяются знания по теме «Движение тела или частиц по параболе». Данный материал вызывает у многих учащихся затруднения, так как теоретических знаний недостаточно. Необходимо иметь практические умения, которые вырабатываются на лабораторных работах и при решении исследовательских задач.

Виртуальные лабораторные работы по физике в настоящее время из-за недостатка оборудования в школах обеспечивают максимальную наглядность, точность соответствия модели реального оборудования для проведения экспериментов. Компьютерная модель позволяет с помощью наблюдений, эксперимента и прочих инструментов исследований собирать самые разные данные, накапливать статическую информацию. Компьютерное моделирование позволяет выявлять зависимости между отдельными факторами, применять статические методы анализа созданной системы.

Модель обычно представляет собой упрощенную модель реальной системы с большим количеством ограничений, что не мешает учащемуся в школе понять и изучить процесс. Актуальность данного проекта заключается в применении виртуальных лабораторных работ в различных видах учебных занятий по физике, что особо актуально в системе очного и дистанционного обучения и позволяет учащимся старших классов посвятить больше сил и времени для углубленной проработки проблемных и важных для них моментов.

Практическая значимость работы заключается в том, что разработанная виртуальная работа может быть использована на уроках учителем и учащимися, а также для подготовки к ЕГЭ.

Объект исследования- движение тела под действием силы тяжести в механике, движение заряженной частицы в электростатическом поле.

Предмет исследования–виртуальные лабораторные работы для наглядного исследования движения по параболе.

Цель моей работы – разработать виртуальные лабораторные работы для наглядного исследования движения тела под действием силы тяжести и движения заряженной частицы в электростатическом поле.

При этом решались следующие задачи:

- Изучить интернет-ресурсы по данной теме;
- Изучить программу Godot Engine;
- Создать упрощенные модели реальной системы с набором ограничений;
- Разработать виртуальные лабораторные работы для наглядного исследования движения тела под действием силы тяжести и движения заряженной частицы в электростатическом поле;
- Разработать образцы лабораторных работ и исследовательских задач;
- Апробировать эффективность созданной программы и сделать выводы.

Были использованы методы:

- Анализ;
- Изучение и обобщение;
- Моделирование;
- Идеализация.

2.Основная часть.

2.1.Исследование интернет - ресурсов.

В настоящее время на российском интернет - пространстве имеется достаточное количество готовых компьютерных программ для проведения виртуальных лабораторных работ по физике. Для того чтобы в полной мере понять возможности созданных виртуальных лабораторий были изучены несколько сайтов с возможностью бесплатного использования данного ресурса. Результаты сведены в таблицу.

Таблица 1. Сравнительный анализ бесплатных виртуальных лабораторных работ.

Ссылка на работу	Плюсы	Минусы
https://fiz.do.am/load/soft/virtualnye_laboratornye_raboty/dvizhenie_tela_broshennogo_pod_uglom_k_gorizontu/29-1-0-82	точность расчётов	программа нормально работает только с определённым набором значений, плохо подходит для лабораторной работы.
http://seninvg07.narod.ru		проблемы с полноэкранным режимом, скорость броска нельзя задать, угол броска имеет только несколько вариантов
http://sverh-zadacha.ucoz.ru/Virtual_lab/9-5/9-5-lab.html		сильно ограничены параметры броска, низкая наглядность
http://virtuallab.by/load/prilozhenie/fizika/dvizhenie_tela_broshennogo_pod_uglom_k_gorizontu_ver_1_0/9-1-0-3	оси на всё расстояние полёта	ограничения по размеру экрана, окно не меняет свой размер, присутствует мерцание, ограниченные параметры броска.
http://mediadidaktika.ru/mod/page/view.php?id=540	точность, разнообразие данных по картинке	AdobeFlashPlayer прекратил работу, программа не запускается

2.2. Godot Engine.

С 8 класса моим основным хобби было создание игр для себя. При поиске игрового движка мной был найден движок Godot Engine, который подходил мне по всем параметрам. Виртуальную лабораторную работу я решил сделать в данной программе, используя язык программирования GDScript.

GodotEngine - это многофункциональный кроссплатформенный игровой движок с унифицированным интерфейсом для создания как 2D-, так и 3D-игр. Он предоставляет полный набор общих инструментов, чтобы пользователи могли сосредоточиться на создании игр без необходимости изобретать колесо. Игры могут быть импортированы в один клик на множество платформ, включая основные настольные платформы (Linux, macOS, Windows), а также мобильные (Android, iOS) и веб-платформы (HTML5).

Godot абсолютно бесплатен и имеет открытый исходный код в соответствии с разрешительной лицензией MIT. Никаких условий, никаких отчислений, ничего. Игры пользователей принадлежат им, вплоть до последней строчки кода движка. Разработка Godot полностью независима и ведется сообществом, что дает пользователям возможность помочь сформировать движок в соответствии со своими ожиданиями. Поддерживается некоммерческой организацией SoftwareFreedomConservancy.(1)

2.3. Моделирование.

Моделирование – метод решения задач, при использовании которого исследуемая система заменяется более простым объектом, описывающим реальную систему и называемым моделью.

Различают физическое и математическое моделирование. Примером физической модели является уменьшенная копия самолета, продуваемая в потоке воздуха. При использовании математического моделирования поведение системы описывается с помощью формул. Особым видом математических моделей являются имитационные модели. Имитационная модель – это компьютерная программа, которая описывает структуру и воспроизводит поведение реальной системы во времени (2).

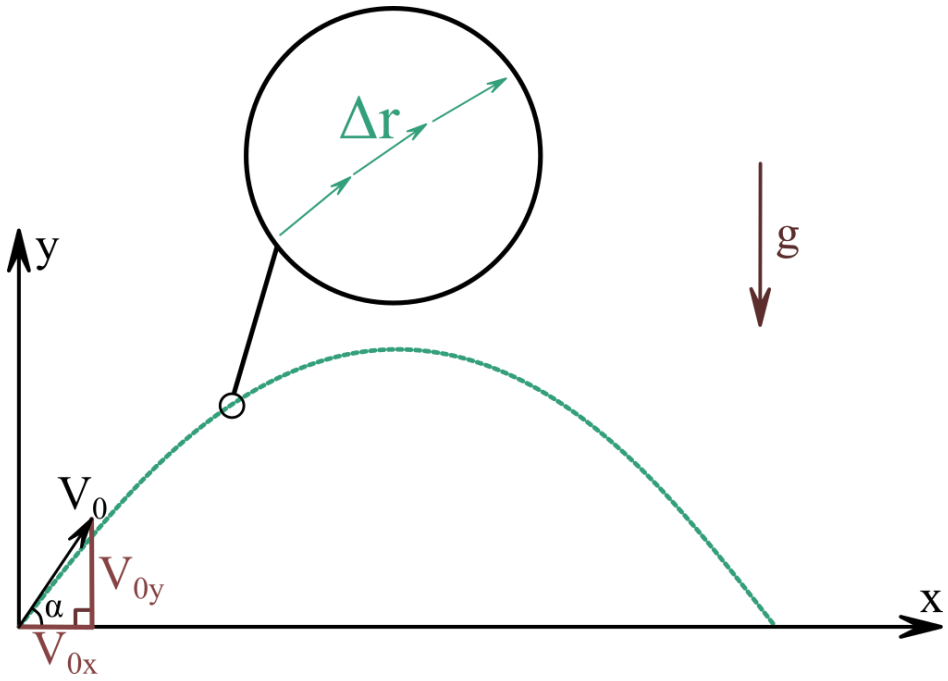
2.3.1. Движение тела под действием силы тяжести.

Вначале мной была рассмотрена реальная система – бросок тела под углом к горизонту. Например, в школе на уроках физической культуры можно наблюдать этот процесс в баскетболе, волейболе, при метании и т.п.

Для имитационной модели мной взяты следующие ограничения:

- Отсутствие силы трения, что дает право брать ускорение свободного падения в пределах 9,5 до 10,5 м/с^2;
- При движении за очень малый промежуток времени можно считать, что тело движется равномерно по формуле $\Delta r = V \cdot \Delta t$, где $\Delta t \to 0$, Δr- вектор перемещения, V - вектор скорости;
- Вектор скорости можно спроецировать на оси и рассматривать в дальнейшем V_x (проекцию на ось x) как постоянную величину, равную $V_0 \cdot \cos\alpha$ (V_0- скорость тела в момент броска, α - угол между горизонтом и вектором скорости теля в момент броска);
- V_y (проекцию скорость на ось y) будет изменяться по формуле $V_y = V_0 \cdot \sin\alpha - g \cdot t$, где g - ускорение свободного падения;
- Оси будут исходить из центра масс тела, как если бы его бросали с земли.

Рисунок 1. Имитационная модель движения тела под действием силы тяжести.



2.3.2. Движение заряженной частицы под действием электрического поля

Движение заряженных частиц широко используется в современных физико-технических установках и приборах: электронный осциллограф, электронный микроскоп, масс-спектрограф и др.

Для имитационной модели мной взяты следующие ограничения:

- Между пластинами конденсатора находится вакуум;
- Сила тяжести частицы настолько мала, что ей можно пренебречь;
- Равнодействующая сил, действующих на частицу равна силе кулона, равной U·q/d, где U - напряжение между пластинами конденсатора, q - заряд частицы, d - расстояние между пластинами.
- Движение частицы в электрическом поле будет рассматриваться как движение тела под действием силы тяжести (см. 3.1), но ускорение свободного падения будет заменено на a = Fкл/m = (U·q)/(d·m)
- Оси будут исходить из края верхней пластины конденсатора, расположенного горизонтально. На нижней пластине конденсатора поддерживается постоянный положительный заряд, на верхней - равный ему по модулю отрицательный.
- Для наглядности придётся выражать все расстояния в мм, скорость в мм/с, заряд частицы в нКл, а её массу в нг.

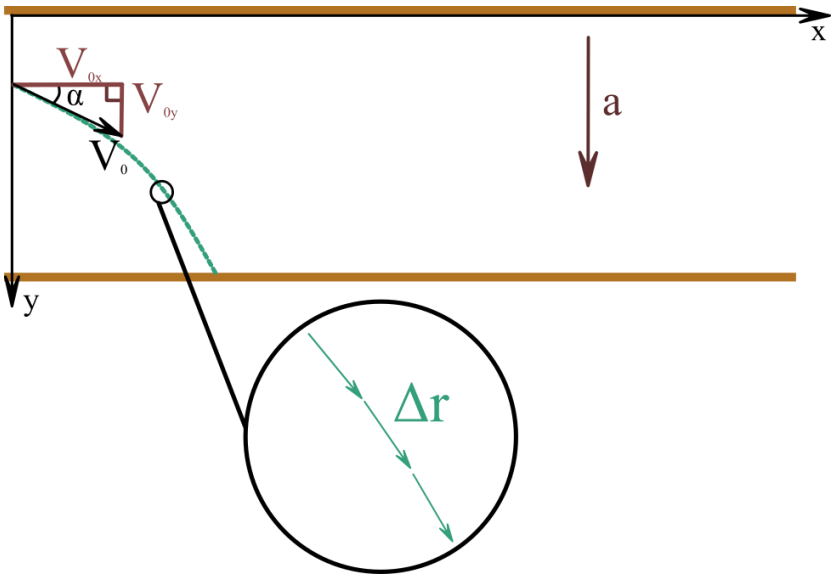


Рисунок 2. Имитационная модель движения заряженной частицы под действием силы электростатического поля.

2.4. Разработка виртуальной лабораторной работы.

2.4.1. Разработка программной модели броска тела.

Godot Engine имеет встроенные классы объектов, которые поддерживают физические взаимодействия. Изначально я попытался использовать их, но мне не удалось настроить их под свои цели, и я сделал перемещение объекта, визуально отмеченного квадратом, через изменение его координат в глобальном сценарии. Все физические величины рассчитываются в СИ.

Координаты объекта изменяются в соответствие с приведёнными выше формулами каждый кадр работы программы после нажатия кнопки "старт", а каждые 0,05 сек. в позиции объекта отрисовывается точка для визуализации траектории. За Δt я взял время между двумя кадрами.

Некоторое время я использовал такую реализацию, но во время одного из тестов мой компьютер немного завис, и из-за увеличения времени между кадрами, траектория искривилась, и полёт был неверно симулирован программой. Данную проблему мне удалось решить, перейдя к использованию метода, который вызывается фиксированное количество раз в секунду. Благодаря этому Δt удалось уменьшить от 1/60 (в лучшем случае) до 1/240 сек, что значительно повысило точность и реалистичность полёта.

Впоследствии я добавил возможность перемещать камеру, чтобы лучше рассмотреть полёт, и справочную информацию.

В программе присутствуют четыре опыта:

- в 1 опыте нет возможности увидеть максимальную координату тела по оси y,
- во 2 опыте не отображается координата по оси x,
- в 3 опыте скрыто время,
- в 4 опыте ускорение свободного падения задаётся случайным образом.

2.4.2. Разработка программной модели полёта частицы.

Полёт частицы в конденсаторе во многом схож с броском тела, поэтому я взял за основу программный код броска. Сделав поправку на единицы измерения, я начал тестировать программу, но частица летела слишком быстро, что снижало наглядность. Чтобы исправить это, мне пришлось за Δt брать не 1/120 сек., а 1/960 сек., не меняя при этом скорость работы программы.

Также для повышения наглядности при высоких скоростях частицы и больших размерах конденсатора, я привязал камеру к частице на время полёта.

Ссылка на программу:

https://cloud.mail.ru/public/aQHU/riwJU4kcw

2.5. Разработка образцов лабораторных работ и исследовательских задач.

Во время планирования лабораторных работ или урока необходимо находить оптимальное соотношение репродуктивных, частичнопоисковых и поисковых работ, чтобы развивать уровень интеллектуальной деятельности. В рамках обучения физики целесообразно использовать виртуальные лабораторные работы, на которых можно предусмотреть самостоятельную деятельность обучающихся. Работы поискового характера дают возможность решать новую для учащихся проблему, опираясь на имеющиеся у них теоретические знания.

Лабораторные работы сопровождаются всегда расчетом погрешности. Многие виртуальные программы дают наглядность проведения опыта с получением одного и тоже результата при определенном наборе данных. Мне удалось за счет встроенной погрешности сделать процесс реалистичным, а виртуальный опыт превратить в лабораторную работу.

На основании разработки можно провести лабораторные работы по следующим темам:

- Нахождение зависимости высоты подъема от скорости
- Нахождение зависимости дальности полета от скорости
- Нахождение зависимости высоты подъема от угла броска
- Нахождение зависимости дальности полета от угла броска
- Нахождение зависимости времени полного полета от скорости
- Нахождение зависимости времени полного полета от угла броска

Перечень можно продолжить, так как у разработанной программы большие возможности изменения параметров.

На уроках можно использовать виртуальную лабораторную работу как исследовательскую задачу. Например, при изучении темы «Ускорение свободного падения» в 9 классе проблемным вопросом может быть оптимальное место для запуска ракет с поверхности Земли. С помощью 3 опыта учащиеся, задавая значения ускорения на разных широтах Земли, получают данные для исследования данной проблемы. Моя виртуальная лабораторная работа может быть одним из видом работ для проверки различных гипотез при изучении тем по механике и электростатике.

3. Заключение.

В результате разработки виртуальной лабораторной работы «Бросок под углом» была достигнута цель и выполнены поставленные задачи. Работая над данным проектом, я пришел к выводу, что учащийся, занимающийся более детально информатикой, математикой , физикой, может создать обучающую программу для бесплатной отработки материала по любым темам.

Десятиклассники апробировали программу для изучения движения под действием силы тяжести. По мнению учащихся, программой легко управлять, она наглядно демонстрирует процесс полета и его реалистичность за счет встроенной погрешности. Им понравилось, что можно моделировать разные условия броска: от вертикально вверх до броска под углом с высоты.

Виртуальную лабораторную работу учитель физики планирует использовать для работы в классах с углубленным изучением физики, а также для нестандартных уроков с элементами исследовательской деятельности.

При дальнейшем обучении в высшем учебном заведении планирую усовершенствовать свои навыки по программированию в области техники.

4. Список источников и литературы.

1. https://docs.godotengine.org/ru/latest/
2. М.С. Эльберг, Н.С.Цыганков. Имитационное моделирование: учебное пособие. Красноярск: Сиб. федер. ун-т, 2017, 128 с.

Приложение 1.

Код программы:

Код меню:

```
extends Control
func _process(delta):
	$help_fade.visible = $help.pressed
func _on_kinematic_throw_btn_pressed():
	return get_tree().change_scene("res://scenes/kinematic_throw.tscn")
func _on_conder_throw_btn_pressed():
	return get_tree().change_scene("res://scenes/conder_throw.tscn")
```

Код броска тела:

```
extends Node2D
var g = Vector2(0, 10.0)
var scl = 48
var y0 = 0
var v0 = 0
var alf = 0
var y_max = 0
var t = 0.0
var t_track = 0.25
var start = false
var end = false
var speed_vector = Vector2(1, 0)
var delay = 0.0
func _ready():
	randomize()
	if Sngltn.exp_num == 1:
		$CanvasLayer/UI/options/y_max_text.hide()
	elif Sngltn.exp_num == 2:
		$CanvasLayer/UI/options/x_text.hide()
	elif Sngltn.exp_num == 3:
		$CanvasLayer/UI/options/t_text.hide()
	elif Sngltn.exp_num == 4:
		$CanvasLayer/UI/options/g.hide()
		$CanvasLayer/UI/options/g/g_edit.text = str(rand_range(9.5, 10.5))
	get_node("CanvasLayer/UI/ExpContainer/Exp" + str(Sngltn.exp_num)).pressed = true
func _physics_process(delta):
	$CanvasLayer/UI/help.visible = $CanvasLayer/UI/help_button.pressed
	$ground.position.x = $obj.position.x
	if start && !end:
		$obj.position /= scl
		t += delta
		t_track += delta
		speed_vector.x = v0 * cos(alf) + rand_range(-0.0005, 0.0005)
		speed_vector.y = -v0 * sin(alf) + g.y * t + rand_range(-0.0005, 0.0005)
		$obj.position += speed_vector * delta
		y_max = max(y_max, max(abs($obj.position.y), y0))
		if $obj.position.y > 0:
			$obj.position.y = 0
			end = true
		$CanvasLayer/UI/options/x_text.text = 'x = ' + str(stepify(($obj.position.x), 0.0001)) + '
м'
		$CanvasLayer/UI/options/y_max_text.text = 'y max = ' + str(stepify(y_max, 0.0001)) + '
м'
```

```
                $CanvasLayer/UI/options/t_text.text = 't = ' + str(stepify(t, 0.01)) + ' c'
                $obj.position *= scl
                if t_track >= 0.05:
                        t_track = 0
                        var new_track = preload("res://objects/track.tscn").instance()
                        new_track.position = $obj.position
                        new_track.rotation = speed_vector.angle()
                        add_child(new_track)
        elif !start && !end:
                y0 = float($CanvasLayer/UI/options/y0/y0edit.text)
                $obj.position.y = - y0 * scl
                $obj/speed_vector.rotation_degrees = -
float($CanvasLayer/UI/options/angle/angl_edit.text)
        delay += delta
        if delay >= 0.01:
                delay = 0
                if $CanvasLayer/UI/left_button.pressed:
                        $Camera2D.position.x -= 4
                if $CanvasLayer/UI/right_button.pressed:
                        $Camera2D.position.x += 4
                if $CanvasLayer/UI/up_button.pressed:
                        $Camera2D.position.y -= 4
                if $CanvasLayer/UI/down_button.pressed:
                        $Camera2D.position.y += 4
                if $CanvasLayer/UI/zoom_in_button.pressed:
                        $Camera2D.zoom -= Vector2(0.01, 0.01)
                if $CanvasLayer/UI/zoom_out_button.pressed:
                        $Camera2D.zoom += Vector2(0.01, 0.01)
func _on_start_button_pressed():
        t = 0.0
        start = true
        $CanvasLayer/UI/start_button.disabled = true
        g.y = float($CanvasLayer/UI/options/g/g_edit.text)
        v0 = float($CanvasLayer/UI/options/V0/V0edit.text)
        alf = deg2rad(float($CanvasLayer/UI/options/angle/angl_edit.text))
        $CanvasLayer/UI/options/V0/V0edit.editable = false
        $CanvasLayer/UI/options/y0/y0edit.editable = false
        $CanvasLayer/UI/options/angle/angl_edit.editable = false
        $CanvasLayer/UI/options/g/g_edit.editable = false
        $obj/speed_vector.hide()
func _on_restart_button_pressed():
        var _err = get_tree().reload_current_scene()
func _on_Exp_pressed(num):
        Sngltn.exp_num = num
        var _err = get_tree().reload_current_scene()
func _on_menu_button_pressed():
        return get_tree().change_scene("res://scenes/menu.tscn")
```

Код полёта частицы в конденсаторе:

```
extends Node2D
var scl = 32
var l = 0
var d = 0
var v0 = 0
var alf = 0
var q = 0
var m = 0
```

```
var u = 0
var y0 = 0
var a = 0
var y_max = 0
var t = 0.0
var t_track = 0.25
var start = false
var end = false
var speed_vector = Vector2(1, 0)
var delay = 0.0
func _ready():
	randomize()
	if Sngltn.exp_conder_num == 1:
		$CanvasLayer/UI/options/u.hide()
		$CanvasLayer/UI/options/u/uedit.text = str(rand_range(5, 220))
	elif Sngltn.exp_conder_num == 2:
		$CanvasLayer/UI/options/q.hide()
		$CanvasLayer/UI/options/q/qedit.text = str(rand_range(1, 5))
	elif Sngltn.exp_conder_num == 3:
		$CanvasLayer/UI/options/alf_text.hide()
	elif Sngltn.exp_conder_num == 4:
		$obj.hide()
		$CanvasLayer/UI/options/y0.hide()
		$CanvasLayer/UI/options/y0/y0edit.text = str(rand_range(d / 2 - d / 4, d / 2 + d / 4))
	get_node("CanvasLayer/UI/ExpContainer/Exp" + str(Sngltn.exp_conder_num)).pressed = true
func _physics_process(delta):
	$CanvasLayer/UI/help.visible = $CanvasLayer/UI/help_button.pressed
	if start && !end:
		$obj.show()
		$obj.position /= scl
		t += delta / 8
		t_track += delta
		#a = Uq/dm
		a = u * q / (d * m)# * pow(10, 6)
		speed_vector.x = v0 * cos(alf) + rand_range(-v0 * 0.01, v0 * 0.01)
		speed_vector.y = v0 * sin(alf) + a * t + rand_range(-v0 * 0.01, v0 * 0.01)
		$obj.position += speed_vector * delta / 8
		y_max = max(y_max, max(abs($obj.position.y), y0))
		$CanvasLayer/UI/options/x_text.text = 'x = ' + str(stepify(($obj.position.x), 0.0001))
		#$CanvasLayer/UI/options/y_max_text.text = 'y max = ' + str(stepify(y_max, 0.0001))
		#$CanvasLayer/UI/options/t_text.text = 't = ' + str(stepify(t, 0.01))
		$obj.position *= scl
		$Camera2D.position = $obj.position
		if $obj.position.y > $conder_minus.position.y:
			$obj.position.y = $conder_minus.position.y
			end = true
		elif $obj.position.y <= 0:
			$obj.position.y = 0
			end = true
		elif $obj.position.x > l * scl:
			#$obj.position.x = l * scl
			end = true
		$CanvasLayer/UI/options/alf_text.text = 'Угол вылета = ' +
str(rad2deg(speed_vector.angle()))
		if t_track >= 0.25:
			t_track = 0
			var new_track = preload("res://objects/track.tscn").instance()
```

```
                new_track.position = $obj.position
                new_track.rotation = speed_vector.angle()
                add_child(new_track)
                move_child(new_track, 0)
        elif !start && !end:
                l = float($CanvasLayer/UI/options/l/ledit.text)
                d = float($CanvasLayer/UI/options/d/dedit.text)
                $conder_minus.polygon[1].x = 144 + l * scl
                $conder_minus.polygon[2].x = 144 + l * scl
                $conder_plus.polygon[1].x = 144 + l * scl
                $conder_plus.polygon[2].x = 144 + l * scl
                $conder_minus.position.y = d * scl
                if Sngltn.exp_conder_num == 4:
                        y0 = rand_range(d / 2 - d / 4, d / 2 + d / 4)
                else:
                        y0 = float($CanvasLayer/UI/options/y0/y0edit.text)
                $obj.position.y = y0 * scl
                $obj/speed_vector.rotation_degrees =
float($CanvasLayer/UI/options/angle/angl_edit.text)
        delay += delta
        if delay >= 0.01:
                delay = 0
                if $CanvasLayer/UI/left_button.pressed:
                        $Camera2D.position.x -= 4
                if $CanvasLayer/UI/right_button.pressed:
                        $Camera2D.position.x += 4
                if $CanvasLayer/UI/up_button.pressed:
                        $Camera2D.position.y -= 4
                if $CanvasLayer/UI/down_button.pressed:
                        $Camera2D.position.y += 4
                if $CanvasLayer/UI/zoom_in_button.pressed:
                        $Camera2D.zoom -= Vector2(0.01, 0.01)
                if $CanvasLayer/UI/zoom_out_button.pressed:
                        $Camera2D.zoom += Vector2(0.01, 0.01)
func _on_start_button_pressed():
        t = 0.0
        #$cube.speed = float($CanvasLayer/UI/options/V0/V0edit.text)
        start = true
        $CanvasLayer/UI/start_button.disabled = true
        v0 = float($CanvasLayer/UI/options/V0/V0edit.text)
        alf = deg2rad(float($CanvasLayer/UI/options/angle/angl_edit.text))
        q = float($CanvasLayer/UI/options/q/qedit.text)
        m = float($CanvasLayer/UI/options/m/medit.text)
        u = float($CanvasLayer/UI/options/u/uedit.text)
        $CanvasLayer/UI/options/l/ledit.editable = false
        $CanvasLayer/UI/options/d/dedit.editable = false
        $CanvasLayer/UI/options/V0/V0edit.editable = false
        $CanvasLayer/UI/options/angle/angl_edit.editable = false
        $CanvasLayer/UI/options/q/qedit.editable = false
        $CanvasLayer/UI/options/m/medit.editable = false
        $CanvasLayer/UI/options/u/uedit.editable = false
        $CanvasLayer/UI/options/y0/y0edit.editable = false
        $obj/speed_vector.hide()
func _on_restart_button_pressed():
        var _err = get_tree().reload_current_scene()
func _on_Exp_pressed(num):
        Sngltn.exp_conder_num = num
```

```
	var _err = get_tree().reload_current_scene()
func _on_menu_button_pressed():
	return get_tree().change_scene("res://scenes/menu.tscn")
```

Код глобального объекта для хранения номера опыта (Sngltn):

```
extends Node
var exp_num = 1
var exp_conder_num = 1
```