Форма № 7

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА
(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій
(повне найменування факультету)

Кафедра механіки, автоматизації та інформаційних технологій
(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «бакалавр»

«Інформаційна технологія керування рухомим об'єктом»
(тема кваліфікаційної роботи українською мовою)
«Information Technology for Managing a Moving Object»
(тема кваліфікаційної роботи англійською мовою)

Виконала: здобувачка денної форми навчання
спеціальності 122 Комп'ютерні науки
(код, назва спеціальності)
Освітня програма «Комп'ютерні науки»
(назва)
Борщ Анастасія Олександрівна
(прізвище, ім'я, по-батькові здобувача)

Керівник к.ф.-м.н., доц. Рачинська А.Л. _____
(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент викл. Недєва О.А.
(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
_____
№ ___ від ____.____. 20___ р.

Завідувач(ка) кафедри
_____ Алла РАЧИНСЬКА
(підпис) (прізвище,ім'я)

Захищено на засіданні ЕК № _____
протокол № __від ____.____.20___ р.

Оцінка_____/____/_____
(за національною шкалою/шкалою ECTS/ бали)
Голова ЕК
_____ Микола МАЛАКСІАНО
(підпис) (прізвище, ім'я)

Одеса 2025

АНОТАЦІЯ

У дипломній роботі розробляється тема «Інформаційна технологія керування рухомим об'єктом».

Мета роботи – створення інформаційної технології для моделювання та керування рухом літального апарату в умовах дії гравітаційного поля з подальшою програмною реалізацією в середовищі тривимірної візуалізації.

У результаті виконання роботи розроблено програмний інструмент з таким функціоналом:

- на основі заданих початкових умов руху та координат об'єкта здійснюється моделювання руху тіла в гравітаційному полі з урахуванням його законів руху;
- реалізовано алгоритм керування рухом літального апарату шляхом поступового переналаштування його вектора швидкості відповідно до цільового напряму, із врахуванням обмежень на зміну модуля швидкості та напрямку;
- розроблено інформаційну технологію дослідження, яка включає тривимірну візуалізацію траєкторії руху, побудову векторів початкової та цільової швидкості, а також анімацію зміни положення та орієнтації об’єкта у просторі;
- програмну реалізацію створено у середовищі Blender, з використанням бібліотек NumPy та mathutils для математичних обчислень і графічного моделювання.

ABSTRACT

The thesis addresses the topic **“Information Technology for Controlling a Moving Object.”**

The aim of the work is to develop an information technology system for simulating and controlling the motion of an object under the influence of a gravitational field, followed by software implementation in a three-dimensional visualization environment.

As a result of the work, a software tool was developed with the following functionality:

• based on specified initial conditions and object coordinates, the motion of the body in a gravitational field is simulated in accordance with the laws of motion;
• an algorithm for controlling the motion of the object was implemented through the gradual adjustment of its velocity vector toward a target direction, taking into account constraints on changes in both the magnitude and direction of the vector;
• an information technology solution was developed, which includes three-dimensional visualization of the motion trajectory, rendering of initial and target velocity vectors, and animation of the object’s position and orientation in space;
• the software implementation was carried out in the Blender environment, using the NumPy and mathutils libraries for mathematical computations and graphical modeling.

ЗМІСТ

стор.

ВСТУП

Актуальність теми. Комп’ютерне моделювання є одним із ключових засобів дослідження складних процесів у механіці та аеродинаміці. У сучасних умовах розвитку інженерних технологій воно широко застосовується у сфері проєктування та оптимізації динамічних систем, зокрема в авіації, космічній техніці, робототехніці та автоматизованих системах керування. Завдяки поєднанню комп’ютерної графіки та чисельних методів стало можливим створення віртуальних середовищ, які імітують реальний рух тіл у просторі, враховуючи вплив зовнішніх сил, таких як гравітація, інерція та керувальні сигнали.

Програми моделювання дозволяють будувати математичні моделі для визначення траєкторій руху, аналізувати поведінку систем у різних умовах, а також тестувати алгоритми керування без ризику для реальних об’єктів. Ефективність таких моделей значною мірою визначається точністю чисельних алгоритмів та відповідністю фізичним умовам.

У зв’язку з цим особливої актуальності набуває розробка інформаційної технології, що дозволяє здійснювати керування рухомим об’єктом (зокрема, літальним апаратом) з урахуванням змін гравітаційного поля та обмежень на зміну швидкості й напрямку.

Використання середовища тривимірної візуалізації Blender у поєднанні з бібліотеками NumPy та mathutils дозволяє реалізувати математичні моделі руху та векторного керування у вигляді динамічної сцени з точною анімацією.

Ціль та задачі дослідження. Метою роботи є розробка інформаційної технології моделювання та керування рухом літального апарату в умовах дії гравітаційного поля з програмною реалізацією алгоритму керування в середовищі тривимірної візуалізації.

Для досягнення поставленої мети необхідно виконати такі задачі:

1. Дослідити фізичні особливості руху об’єктів у гравітаційному полі.

2. Побудувати математичну модель алгоритму керування рухом літального апарату з обмеженням на зміну модуля та напрямку вектора швидкості.
3. Реалізувати чисельний алгоритм корекції вектора швидкості відповідно до цільового значення.
4. Змоделювати віртуальну сцену у Blender, яка відображає динаміку об’єкта, вектори швидкості та напрямку.
5. Провести дослідження поведінки системи при різних початкових умовах і параметрах, оцінити точність та ефективність розробленого алгоритму.

Об’єкт дослідження – рух літального апарату в умовах гравітаційного поля.

Предмет дослідження – вектор швидкості літального апарату.

1 РУХ ТІЛА В ГРАВІТАЦІЙНОМУ ПОЛІ

1.1 Рух тіла в полі сил тяжіння

Розглянемо рівняння руху важкої матеріальної точки у безповітряному просторі [1]. Нехай матеріальна точка рухається в однорідному полі тяжіння під дією лише однієї сили тяжіння mg, сталої за числовим значенням і напрямком. Знайдемо рівняння руху точки в декартовій системі координат, вважаючи вісь z напрямленою паралельно силі по вертикалі вгору (див. рис. 1.1). Диференціальні рівняння руху мають вигляд:

$$\begin{cases} m\ddot{x} = 0, \\ m\ddot{y} = 0, \\ m\ddot{z} = -mg. \end{cases} \tag{1.1}$$

Інтегруючи рівняння (1.1), отримаємо три перших інтеграли:

$$\begin{cases} \dot{x} = c_1, \\ \dot{y} = c_2 \\ \dot{z} = -gt + c_3. \end{cases} \tag{1.2}$$

Інтегруючи ще раз, знайдемо загальне розв’язання:

$$\begin{cases} x = c_1 t + c_4, \\ y = c_2 t + c_5 \\ z = -\frac{gt^2}{2} + c_3 t + c_6. \end{cases} \tag{1.3}$$

Отримані рівняння дають в залежності від значень довільних сталих $c_1, c_2, \dots, c_6$ цілий клас рухів. У кожному конкретному випадку ці сталі можна визначити, знаючи початкові умови, і знайти, таким чином, закони відповідного руху.

Розглянемо два часткові випадки руху важкої матеріальної точки у безповітряному просторі при різних початкових умовах. Рух важкої матеріальної точки, кинутої вертикально вгору.

Нехай точка в початковий момент знаходиться в початку координат і має швидкість v_0, напрямлену вертикально вгору. Цій умові відповідають початкові умови:

$$\text{При } t = 0 \left| \begin{matrix} x=y=z=0, \\ \dot{x}=\dot{y}=0, \dot{z}=v_0. \end{matrix} \right.$$

Підставляючи ці початкові дані у рівняння (1.2) і (1.3), визначимо довільні сталі:

$$c_1 = c_2 = 0,$$
$$c_3 = v_0,$$
$$c_4 = c_5 = c_6 = 0.$$

І будемо мати для цього випадку закон руху:

$$\begin{cases} x = 0, \\ y = 0, \\ z = v_0 t - \frac{gt^2}{2}. \end{cases} \qquad (1.4)$$

тобто рух буде прямолінійним; він буде рівномірно сповільненим до моменту, коли швидкість $\dot{z} = v_0 - gt$ стане рівною нулю, а потім - рівномірно прискореним.

Наступний випадок - рух важкої матеріальної точки, кинутої під кутом до горизонту. Нехай точка в початковий момент знаходиться в початку координат і має швидкість v_0, яка лежить у площині Oxz і напрямлена під кутом α до горизонту (рис. 1). У цьому випадку початкові умови будуть:

при $t = 0 \left|_{\dot{x}=v_0 \cos\alpha,\ \dot{y}=0,\ \dot{z}=v_0 \sin\alpha}^{x=y=z=0,}\right.$.

Підставляючи ці початкові дані у рівняння (1.2) і (1.3), знайдемо для постійних інтегрування значення:

$$c_1 = v_0 \cos\alpha,$$
$$c_2 = v_0 \sin\alpha,$$
$$c_3 = 0,$$
$$c_4 = c_5 = c_6 = 0.$$

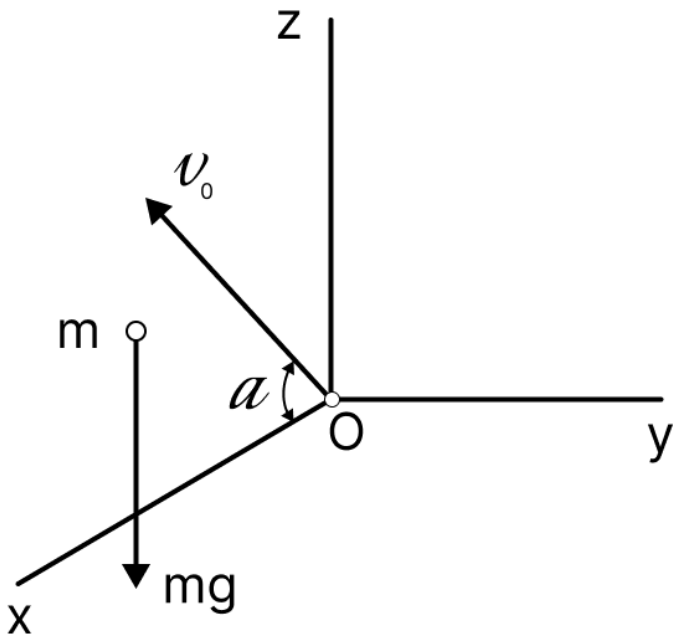

Рисунок 1.1 - Траєкторія руху тіла, кинутого під кутом α до горизонту

Отже, закон руху точки визначається рівняннями:

$$\begin{cases} x = v_0 t \cos\alpha, \\ y = 0, \\ z = v_0 t \sin\alpha - \frac{1}{2} g t^2. \end{cases} \qquad (1.5)$$

Із цих рівнянь видно, що траєкторія точки буде деякою кривою, яка лежить у площині Oxz. Незалежно від рівнянь (1.5), це твердження випливає безпосередньо з того, що напрям сили, що діє на точку, постійний.

Знайдемо рівняння цієї траєкторії в не параметричній формі. Для цього виключимо t з рівнянь (1.5). Із першого рівняння отримаємо:

$$t = \frac{x}{v_0 \cos \alpha};$$

підставляючи цей вираз в останнє рівняння, знайдемо:

$$z = x \tan \alpha - \frac{g x^2}{2 {v_0}^2 \cos^2 \alpha}. \qquad (1.6)$$

Як видно, траєкторія є параболою з віссю, паралельною осі z, яка проходить через початок координат. Кут α, який утворює початкова швидкість з горизонтальною площиною, називається кутом кидання.

Дослідимо деякі властивості розглядуваного руху

а) Знайдемо горизонтальну дальність польоту, тобто відрізок

OA (рис. 1.2). Для цього визначимо точки перетину траєкторії (1.6) з віссю Ox. Припускаючи $z = 0$, отримаємо:

$$x \left(\tan \alpha - \frac{g x}{2 {v_0}^2 \cos^2 \alpha} \right) = 0.$$

Звідси знайдемо дві точки перетину: $x_1 = 0$, (тобто початок координат) та $x_2 = OA = \frac{2 {v_0}^2 \tan \alpha \cos^2 \alpha}{g} = \frac{{v_0}^2 \sin 2\alpha}{g}$. (1.7)

Це і буде шуканою горизонтальною дальністю.

Очевидно, що при даній початковій швидкості v_0 вона буде найбільшою, коли:

$$\sin 2\alpha = 1,$$

тобто при $\alpha = \frac{\pi}{4}$. Зауважимо, що при куті кидання $\alpha = 45°$ точка матиме найбільшу горизонтальну дальність у безповітряному просторі; в повітрі цей кут буде дещо меншим.

Так як:

$$\sin 2\alpha = \sin(\pi - 2\alpha),$$

то, поклавши $\pi - 2\alpha = 2\beta$, знайдемо, що для кута $\beta = \frac{\pi}{2} - \alpha$ маємо:

$$\sin 2\alpha = \sin 2\beta,$$

звідки видно, що при кутах кидання

α і $\beta = 90° - \alpha$ горизонтальна дальність буде однакова (див. рис. 1.2). Інакше кажучи, горизонтальна дальність буде однакова незалежно від того, направимо ми початкову швидкість під кутом α до горизонту чи під кутом β до вертикалі (настильна і навісна траєкторії).

б) Знайдемо тепер максимальну висоту підйому точки при даному куті кидання α; для цього потрібно знайти максимум z, тобто прирівняти до нуля похідну $\frac{dz}{dx}$. Використовуючи рівняння (1.6), отримаємо:

$$\frac{dz}{dx} = \tan\alpha - \frac{gx}{v_0{}^2 \cos^2\alpha} = 0,$$

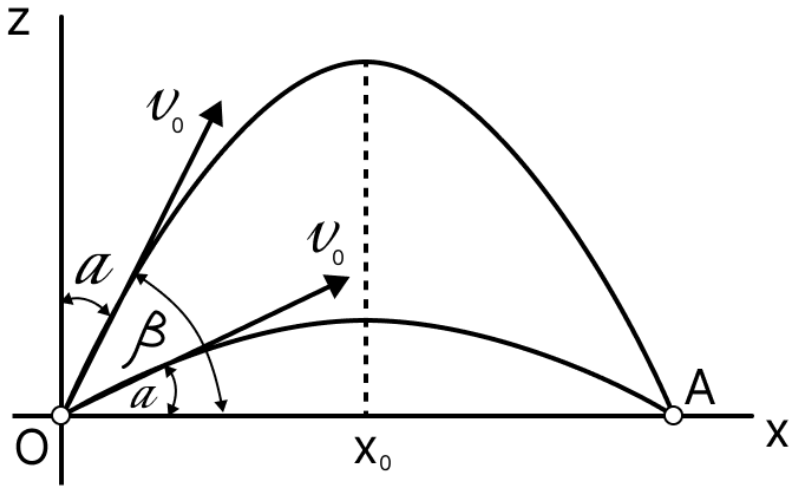

Рисунок 1.2 – Траєкторія кидання під кутом α та β, максимальна висота та горизонтальна дальність

Звідси випливає, що z буде мати максимум при:

$x_0 = \frac{v_0{}^2 \tan\alpha \cos^2\alpha}{g} = \frac{v_0{}^2 \sin\alpha \cos\alpha}{g}$, (що z має максимум, видно по знаку $\frac{d^2z}{dx^2}$).

Підставляючи це значення x_0 у рівняння (1.6), знайдемо:

$$z_{max} = \frac{{v_0}^2 \sin\alpha \cos\alpha \tan\alpha}{g} - \frac{g{v_0}^4 \sin^2\alpha \cos^2\alpha}{2g^2{v_0}^2 \cos^2\alpha},$$

або, після спрощення:

$$H = z_{max} = \frac{{v_0}^2}{2g}\sin^2\alpha. \qquad (1.8)$$

Звідси видно, що при даній початковій швидкості v_0 найбільша висота підйому буде, коли $\alpha = \frac{\pi}{2}$, тобто коли початкова швидкість напрямлена вертикально вгору. У цьому випадку:

$$z_{max} = \frac{{v_0}^2}{2g} \qquad (1.9)$$

в) Знайдемо, час руху T із точки O до точки A. Підставляючи в перше з рівнянь (5) замість x величину горизонтальної дальності x_2, визначену рівністю (7), отримаємо:

$$T = \frac{2v_0}{g}\sin\alpha.$$

З попереднього видно, що для різних кутів кидання при однаковій початковій швидкості ми будемо мати траєкторії руху точок у вигляді парабол, що лежать у площині Oxz, причому для граничних парабол:

$$x_{max} = \frac{{v_0}^2}{g},\ z_{max} = \frac{{v_0}^2}{2g}.$$

Якщо побудувати огинаючу сімейство парабол (1.6), то ми отримаємо деяку криву, яка обмежить ту частину площини, куди може потрапити точка, якщо їй надано певна початкова швидкість v_0, під яким би кутом до горизонту остання не була направлена. Знайдемо рівняння цієї кривої.

Для знаходження огинаючої сімейства кривих

$z = f(x, \alpha)$ потрібно використати параметр α як змінну:

$z = f(x, \alpha)$ і $\frac{\partial z}{\partial \alpha} = f'(x, \alpha) = 0.$

Для зручності приймемо в рівнянні (1.6) змінним параметром не кут α, а $p = \tan \alpha$; тоді, зауважимо, що: $\frac{1}{\cos^2 \alpha} = \sec^2 \alpha = 1 + \tan^2 \alpha = 1 + p^2$, отримаємо рівняння (1.6) у вигляді:

$$z = px - \frac{gx^2}{2{v_0}^2}(1 + p^2). \qquad (1.10)$$

Приєднаємо до нього рівняння:

$$\frac{\partial z}{\partial p} = x - \frac{pgx^2}{{v_0}^2} = 0 \qquad (1.11)$$

і виключимо з рівнянь (1.10) і (1.11) параметр p. Із рівняння (1.11) отримаємо:

$$p = \frac{{v_0}^2}{gx}.$$

Підставляючи це значення p в (1.10), після спрощень отримаємо рівняння огинаючої у вигляді:

$$z = \frac{{v_0}^2}{2g} - \frac{g}{2{v_0}^2}x^2 \ . \qquad (1.12)$$

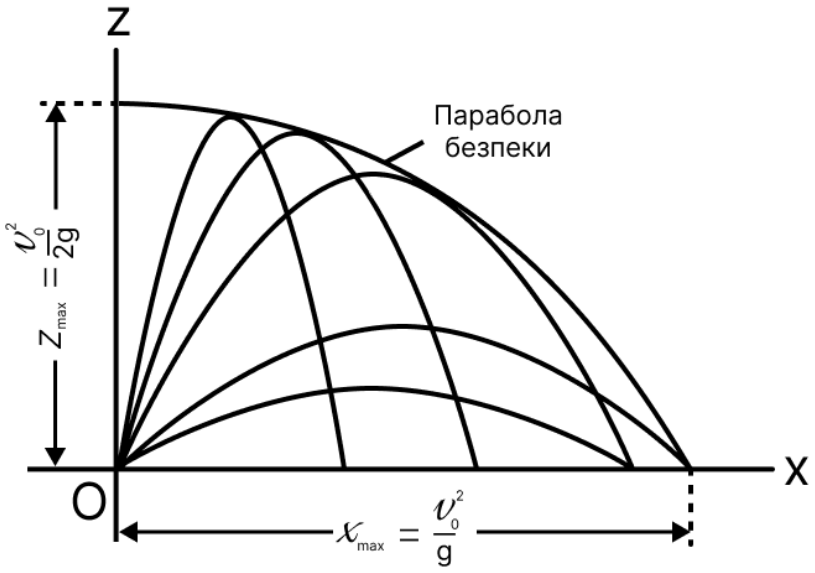

Рисунок 1.3 - Парабола безпеки та параболічні траєкторії для різних кутів кидання при однаковій швидкості

Таким чином, огинаюча буде параболою з віссю, паралельною осі z (рис. 3) Ця парабола називається параболою безпеки. Легко побачити, що вона проходить через точки найбільшої дальності та найбільшої висоти підйому. Дійсно, поклавши $z = 0$, отримаємо з рівняння (1.12):

$x^2 \frac{g}{2{v_0}^2} = \frac{{v_0}^2}{2g}$, звідки $x = \frac{{v_0}^2}{g} = x_{max}$

при $x = 0 \Rightarrow$

$z = \frac{{v_0}^2}{2g} = z_{max}$.

1.2 Рух тіла під дією центральних сил

Розглянемо рух вільної матеріальної точки під дією центральних сил [2]. За законом площ, якщо діюча на точку сила є центральною і початок координат взято в центрі, через який проходить лінія дії сили, то $r \times F = 0$. Тоді теорема про зміну моменту кількості руху:

$$\frac{d}{dt}(r \times mv) = r \times F$$

дає перший інтеграл:

$$r \times v = c \tag{1.13}$$

Звідси випливає, що траєкторія точки, яка рухається під дією центральної сили, є плоскою кривою, а сам рух відбувається за законом площ, тобто з постійною секторною швидкістю, або, іншими словами, радіус-вектор точки, проведений з центру сили, за будь-які рівні проміжки часу описує рівні площі.

Із рівняння (1.13) знаходимо, що закон площ можна виразити рівнянням:

$$|r \times v| = 2\frac{d\sigma}{dt} = c, \tag{1.14}$$

де $\frac{d\sigma}{dt}$ - це секторна швидкість точки, а c - стала, яка називається сталою площ.

Значення c визначається початковими даними. Якщо в початковий момент $r = r_0$ і $v = v_0$, то:

$$c = |mom_0(v_0)| = r_0 v_0 \sin(\widehat{r_0, v_0}) \tag{1.15}$$

Оскільки при русі під дією центральної сили траєкторія точки є плоскою кривою, для вивчення руху зручно користуватися полярними координатами r і φ, що значно спрощує розрахунки.

У полярних координатах:

$$\frac{d\sigma}{dt} = \frac{1}{2} r^2 \frac{d\varphi}{dt},$$

і рівняння (1.14), яке виражає закон площ, набуде вигляду:

$$r^2 \frac{d\varphi}{dt} = c. \tag{1.16}$$

Швидкість матеріальної точки, що рухається під дією центральної сили описана нижче. Відомо, що в полярних координатах r і φ швидкість точки виражається формулою:

$$v^2 = \left(\frac{dr}{dt}\right)^2 + r^2 \left(\frac{d\varphi}{dt}\right)^2, \tag{1.17}$$

де:

$$\frac{dr}{dt} = v_r \text{ і } r \frac{d\varphi}{dt} v_p . \tag{1.18}$$

Суть - відповідно радіальна та трансверсальна (кутова) проєкції швидкості. Перетворимо вирази (1.18), виключивши з них час t за допомогою рівняння (1.18). Тоді отримаємо:

$$v_r = \frac{dr}{dt} = \frac{dr}{d\varphi}\frac{d\varphi}{dt} = \frac{c}{r^2}\frac{dr}{d\varphi},$$

$$v_p = r\frac{d\varphi}{dt} = \frac{c}{r}.$$

Введемо нову змінну:

$$u = \frac{1}{r}. \tag{1.19}$$

Тоді, враховуючи, що:

$$\frac{du}{d\varphi} = -\frac{1}{r^2}\frac{dr}{d\varphi}, \tag{1.20}$$

отримуємо:

$$v_r = -c\frac{du}{d\varphi},\ v_p = cu \tag{1.21}$$

І остаточно знайдемо для швидкості точки, що рухається під дією центральної сили, вираз:

$$v^2 = c^2\left[\left(\frac{du}{d\varphi}\right)^2 + u^2\right]. \tag{1.22}$$

Розглянемо диференціальні рівняння руху точки під дією центральної сили та використання формули Біне. Щоб отримати названі рівняння, звернемося до теореми про зміну кінетичної енергії точки. У випадку центральної сили (див. рис. 1.4) елементарна робота: $F \cdot dr = F_r dr$, де $F_r = F$- для сил тяжіння. Отримаємо:

$$d\left(\frac{mv^2}{2}\right) = F_r dr,$$

або, поділивши обидві частини рівності на $d\varphi$:

$$\frac{m}{2}\frac{d(v^2)}{d\varphi} = F_r\frac{dr}{d\varphi}.$$

Підставляючи сюди замість v^2 рівність (1.22), отримаємо:

$$\frac{c^2 m}{2}\frac{d}{d\varphi}\left[\left(\frac{du}{d\varphi}\right)^2 + u^2\right] = F_r \frac{dr}{d\varphi}$$

Враховуючи, що згідно рівності (1.20), $\frac{dr}{d\varphi} = -r^2\frac{du}{d\varphi} = -\frac{1}{u^2}\frac{du}{d\varphi}$, отримаємо:

$$\frac{m}{2}c^2\left[2\frac{du}{d\varphi}\frac{d^2u}{d\varphi^2} + 2u\frac{du}{d\varphi}\right] = F_r\frac{1}{u^2}\frac{du}{d\varphi}\,.$$

Скорочуючи обидві частини на $\frac{du}{d\varphi}$ та домноживши на u^2 , отримаємо остаточну форму:

$$mc^2u^2\left(\frac{d^2u}{d\varphi^2} + u\right) = -F_r. \qquad (1.23)$$

Це рівняння називається формулою Біне (Binet).

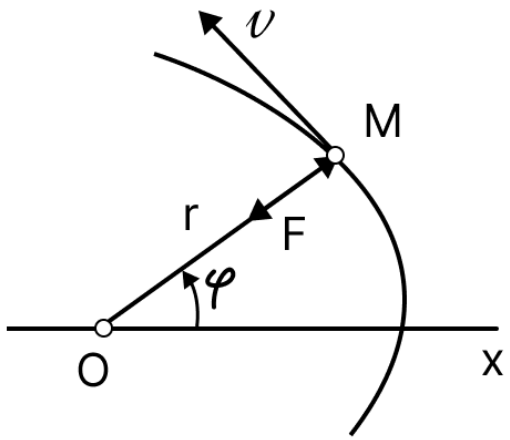

Рисунок 1.4 - Схема дії центральної сили, вектор швидкості v, радіус-вектор r, кут φ.

У загальному випадку: $F_r = F_r(r, \varphi, v_r, v_p, t)$ або, як видно з (1.19) та (1.20) $F_r = F_r\left(u, \varphi, \frac{du}{d\varphi}, t\right)$.

Приєднаємо сюди рівняння (1.16), яке виражає закон площ:

$$\frac{d\varphi}{dt} = cu^2 , \qquad (1.24)$$

Маємо систему з двох диференціальних рівнянь (1.23) і (1.24), з яких, знаючи F_r, можна знайти залежності r та φвід t, тобто знайти закон руху точки під дією центральної сили.

Особливий інтерес становить випадок, коли сила Fявно не залежить від часу. Тоді рівняння (1.23), яке зв'язує F_r з координатами r і φ, буде представляти собою диференціальне рівняння траєкторії точки. З нього можна безпосередньо визначити, під дією якої центральної сили точка може описувати дану траєкторію, і, навпаки, знайти, яку траєкторію точка опише під дією заданої центральної сили.

Закон руху точки вздовж траєкторії знайдеться при цьому з рівняння (1.24).

Отримані рівняння відіграють важливу роль при вивченні руху в полі тяжіння Сонця або планет (небесна механіка, динаміка ракет, космонавтика).

Розглянемо простіший приклад. Нехай під дією центральної сили точка рухається по колу $r = a = const$, де a - радіус. Підставляючи це в формулу Біне, отримаємо:

$$F_r = -mc^2u^3 = -\frac{mv^2}{a^3}$$

Отже, сила за модулем є сталою. Із формули (1.22) також видно, що чисельна величина швидкості точки в цьому випадку також стала й дорівнює:

$$v = cu = \frac{c}{a}.$$

Виключаючи з двох отриманих рівностей сталу c, знайдемо:

$$F_r = -\frac{mv^2}{a}$$

Таким чином, рух вільної матеріальної точки маси m по колу радіуса a відбувається з сталою швидкістю $v = v_0$ під дією сталої притягальної сили, рівної $\frac{mv^2}{a}$. Цю силу також називають доцентровою силою.

Тоді розглянемо рух планет та закон всесвітнього тяжіння. В основі небесної механіки лежать три закони, відкриті Кеплером (1571-1630). Ці закони були отримані з численних спостережень астронома Тихо Браге за рухом планет і полягають у наступному:

1) Усі планети (і комети) рухаються навколо Сонця по плоских орбітах, дотримуючись закону площ.

2) Орбіти - це конічні перерізи, в одному з фокусів яких знаходиться Сонце.

3) Квадрати зоряних періодів обертання планет навколо Сонця пропорційні кубах великих півосей їхніх орбіт.

На основі законів Кеплера Ньютон сформулював закон, згідно з яким змінюється сила, що діє на планету під час її руху навколо Сонця, і таким чином прийшов до закону всесвітнього тяжіння.

Покажемо, як може бути вирішена задача динаміки, що полягає в тому, щоб, знаючи закон даного руху (закони Кеплера), визначити діючу силу. Із першого закону Кеплера безпосередньо випливає, що сила, яка діє на планети, є центральною - напрямок якої проходить через центр Сонця. Із другого закону легко знайти, що ця сила, яка діє на планети, є притягальною до Сонця і обернено пропорційна квадрату відстані. Для цього скористаємося формулою Біне.

Як відомо, рівняння конічного перерізу в полярних координатах має вигляд:

$$r = \frac{p}{1+e\cos\varphi} \text{ або } a = \frac{1+e\cos\varphi}{p}, \tag{1.25}$$

де: e - ексцентриситет, p - параметр, причому в разі еліпса або гіперболи: $p\frac{b^2}{a}$

де: a і b - відповідно велика та мала напівосі.

Так як за другим законом Кеплера орбіта є конічним перерізом, то, підставивши з рівняння (1.25) значення u в формулу Біне, знайдемо діючу силу.

Виконавши підстановку, отримаємо:

$$\frac{mc^2u^2}{2}[-e\cos\varphi + 1 + \mathrm{e}\cos\varphi] = -F_r\ ,$$

звідки: $F_r = \frac{c^2mu^2}{p}$.

Введемо позначення:

$$\frac{c^2}{p} = \mu, \tag{1.26}$$

де μ називається сталою Гаусса. Тоді, оскільки $u = \frac{1}{r}$, отримаємо:

$$F_r = -\mu\frac{m}{dr^2} \tag{1.27}$$

Таким чином, діюча сила F буде силою притягання, що змінюється обернено пропорційно квадрату відстані від притягуючого центру.

З третього закону Кеплера випливає, що стала μ буде однаковою для всіх тіл Сонячної системи. Дійсно, третій закон Кеплера можна подати у вигляді:

$$\frac{a^3}{T^2} = const \text{ або } \frac{4\pi^2 a^3}{T^2} = \text{const.} \qquad (1.28)$$

Згідно з рівністю (1.14), стала площ c дорівнює подвоєній секторній швидкості, тобто подвоєному відношенню описаної радіус-вектором площі до відповідного часу. Оскільки площа еліпса дорівнює πab, то в нашому випадку: $c = \frac{2\pi ab}{T}$ і $c^2 = \frac{4\pi^2 a^2 b^2}{T^2}$.

Вводячи параметр $p = \frac{b^2}{a}$, отримаємо: $c^2 = \frac{4\pi^2 a^3 p}{T^2}$, звідки: $\frac{4\pi^2 a^3}{T^2} = \frac{c^2}{p}$.

Але за попереднім $\frac{c^2}{p} = \mu$, і ми, згідно з (1.28), отримаємо:

$$\mu = \frac{4\pi^2 a^3}{T^2} = const. \qquad (1.29)$$

Отже, коефіцієнт μ (стала Гаусса) є величиною сталою для всіх тіл, які рухаються під дією притягання Сонця, тому вона повинна залежати тільки від маси Сонця.

З попереднього легко вивести відкритий Ньютоном закон всесвітнього тяжіння. Для тіл, які рухаються під дією притягання Землі, існує своя стала Гаусса. Назвемо її λ. Сила, з якою Сонце притягує Землю:

$$F_{r1} = -\frac{\mu m}{r^2}, \qquad (1.30)$$

де m - маса Землі. Сила, з якою Земля притягує Сонце:

$$F_{r2} = -\frac{\lambda M}{r^2}, \qquad (1.30')$$

де M - маса Сонця. За законом дії і протидії маємо: $F_{r1} = F_{r2}$ або $\frac{\mu m}{r^2} = \frac{\lambda M}{r^2}$, звідки: $\frac{\mu}{M} = \frac{\lambda}{m} = \frac{\lambda_n}{m_n} = const$, де $\frac{\lambda_n}{m_n}$ - відношення сталої Гаусса будь-якої планети до її маси.

Це відношення є сталою величиною, яка називається гравітаційною сталою. Позначимо її літерою f. Тоді:

$$\frac{\mu}{M} = \frac{\lambda}{m} = f,$$

звідки: $\mu = fM, \lambda = fm$.

Підставляючи це значення в рівняння (1.30) або (1.30'), і позначаючи $|F_{r1}| = |F_{r2}| = F$ отримаємо:

$$F = f\frac{Mm}{r^2}. \qquad (1.31)$$

Ця формула виражає закон всесвітнього тяжіння: два тіла притягуються силою, прямо пропорційною добутку їх мас і обернено пропорційною квадрату відстані між ними.

Розмірність гравітаційної сталої в абсолютній системі:

$$[f]\frac{M \cdot L \cdot L^2}{T^2 M^2} = \frac{L^3}{MT^2}$$

У системі одиниць (SI): $f = 6{,}673 \cdot 10^{-11}$ м^3/кгсек^2.

При визначені сили, що діє на точку, яка рухається за законами Кеплера, ми брали рівняння конечного перерізу у вигляді (1.25), враховуючи, що за другим законом Кеплера центр тяжіння знаходиться в одному з фокусів.

Можна вирішити більш загальну задачу: знайти закон центральної сили, що залежить тільки від положення точки, під дією якої точка при

певних початкових умовах описує деякий конічний переріз. У такому вигляді задача була вирішена Бертраном, який виявив, що сила в цьому випадку буде притягальною або прямо пропорційною відстані до центру, або обернено пропорційною квадрату цієї відстані.

Кеніг поставив задачу ще ширше, а саме: знайти закон сили $f(r)$, під дією якої точка описує алгебраїчну криву при будь-яких початкових умовах. Рішення цієї задачі привело Кенігса до такого ж результату.

Спостереження за подвійними зорями показують, що зоря-супутник рухається навколо головної зорі по еліпсу, в фокусі якого знаходиться головна зоря. Відповідно, тут має місце ньютонівський закон тяжіння. Якби сила була пропорційна відстані, то головна зоря перебувала б у центрі орбіти супутника, що суперечить спостереженням.

Знайдемо траєкторію матеріальної точки, що притягується нерухомим центром із силою, обернено пропорційною квадрату відстані (задача Ньютона).

Для вирішення цієї задачі скористаємося формулою Біне, поклавши: $F_r = -\frac{\mu m}{r^2} = -\mu m u^2$;

Отримаємо рівняння:

$$mc^2u^2\left(\frac{d^2u}{d\varphi^2} + u\right) = \mu m u^2,$$

або, поділивши обидві частини на mc^2u^2, маємо:

$$\frac{d^2u}{d\varphi^2} + u = \frac{\mu}{c^2}. \qquad (1.32)$$

Загальне розв'язання цього рівняння: $u = \frac{\mu}{c^2} + \alpha\cos(\varphi + \varepsilon)$, де α і ε - сталі інтегрування.

Вводячи відповідно до рівності (1.26) позначення:

$$p = \frac{c^2}{\mu}. \tag{1.33}$$

і виносячи множник $\frac{\mu}{c^2} = \frac{1}{p}$ за дужки, вводимо нову сталу $e = \alpha p$, замість α отримаємо:

$$u = \frac{1+e\cos(\varphi+\varepsilon)}{p}. \tag{1.34}$$

Порівнюючи результат із рівнянням (1.25), бачимо, що траєкторія точки буде конічним перерізом (еліпс, парабола або гіпербола), один із фокусів якого збігається з притягуючим центром О.

Конкретний вид траєкторії залежить від початкових умов - значень e, ε тощо.

Зазначимо, що вибір початку відліку кута φ, тобто полярної осі, можна зробити $\varepsilon = 0$. Тоді:

$$u = \frac{1+e\cos\varphi}{p} \text{ або } r = \frac{p}{1+e\cos\varphi}. \tag{1.35}$$

Оскільки при $\varphi = 0$ косинус має найбільше значення, отже, вважаючи $\varepsilon = 0$, ми визначаємо кут φ від тієї точки траєкторії, для якої u - максимум, а r - мінімум, тобто від точки P орбіти, найближчої до притягуючого центру (рис. 1.5), і називаємо її перицентром (від грец. περί - поблизу). При русі навколо Сонця перицентр називається перигелієм, а навколо Землі - перигеєм. Протилежна точка називається апогелієм чи апогеєм.

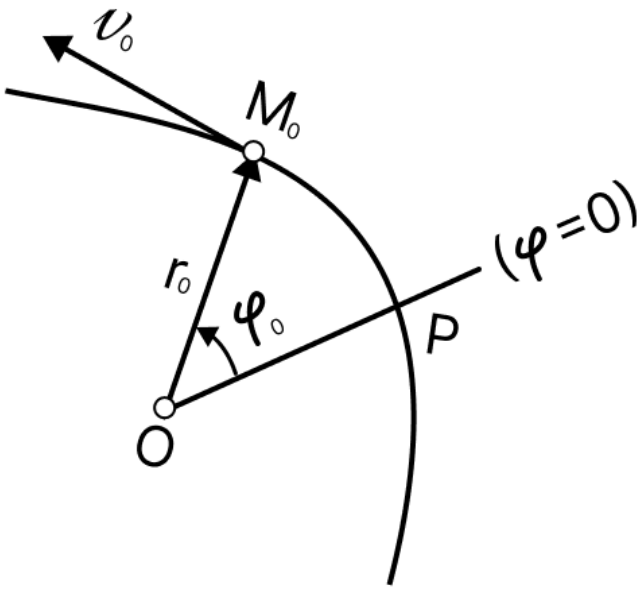

Рисунок 1.5 – Схема найближчої орбіти до притягуючого центру

Положення цієї точки в майбутньому - незмінне й визначається початковими даними.

З рівняння (1.35) маємо:

$$\frac{du}{d\varphi} = -\frac{e}{p}\sin\varphi. \qquad (1.36)$$

Нехай у початковому положенні M_0 точка знаходиться на відстані r_0 від притягуючого центру й має початкову швидкість v_0. При цьому $POM_0 = \varphi_0$ є визначальним кутом, що показує положення перицентра P відносно початкового положення точки M_0.

Тоді, знаходячи перше значення $\frac{du}{d\varphi}$, з рівняння (1.22) матимемо початкові умови:

$$\text{При } \varphi = \varphi_0 \; u = u_0 = \frac{1}{r_0}, \left(\frac{du}{d\varphi}\right)_0 = -\sqrt{\frac{{v_0}^2}{c^2} - {u_0}^2}. \qquad (1.37)$$

Беремо знак мінус перед коренем, враховуючи, що швидкість v_0 напрямлена так, що знаки v_{r0} і $v_{\varphi 0}$, а відповідно, і $(dr)_0$, $(d\varphi)_0$ однакові;

Тоді, як видно з рівності (1.20) має бути $\left(\frac{du}{d\varphi}\right)_0 < 0$.

Підставляючи початкові дані (1.37) у рівняння (1.37) та (1.36), отримаємо: $u_0 = \frac{1+e\cos\varphi}{p}$, $\frac{1}{c}\sqrt{{v_0}^2 - c^2{u_0}^2} = \frac{e}{p}\sin\varphi_0$.

Замінивши p його значенням з формули (1.33), отримаємо:

$$\mathrm{e}\sin\varphi_0 = \frac{c}{\mu}\sqrt{{v_0}^2 - c^2{u_0}^2}, \quad \mathrm{e}\cos\varphi_0 = \frac{c^2}{\mu}u_0 - 1. \tag{1.38}$$

Звідси, ділячи перше рівняння на друге та підносячи обидва до квадрата і складаючи, отримаємо:

$$\tan\varphi_0 = \frac{c\sqrt{{v_0}^2 - c^2{u_0}^2}}{c^2u_0 - \mu}. \tag{1.39}$$

$$e = \sqrt{1 + \frac{c^2}{\mu^2}({v_0}^2 - 2\mu u_0)}. \tag{1.40}$$

Значення сталої площі c обчислюється за початковими умовами згідно з формулою (1.15).

Формула (1.39) дає значення кута φ_0 , яке визначає положення перицентра траєкторії щодо початкового радіус-вектора r_0. Ексцентриситет e визначається з формули (1.40). Як видно, значення e залежить від знака величини

$$h = {v_0}^2 - 2\mu u_0 = {v_0}^2 - \frac{2\mu}{r_0} \tag{1.41}$$

Встановимо фізичний сенс цієї величини [3]. Приймаючи, що потенціальна енергія V точки в полі тяжіння визначається формулою $V = -\frac{\mu m}{r}$, вичислимо повну початкову енергію цієї точки. Отримаємо:

$$E_0 = \frac{m{v_0}^2}{2} + V_0 = \frac{m{v_0}^2}{2} - \frac{\mu m}{r} = \frac{m}{2}\left({v_0}^2 - \frac{2\mu}{r_0}\right).$$

Отже, h- величина, пропорційна повній початковій енергії точки. Тип траєкторії залежить від знака цієї енергії:

якщо $h < 0$, тобто ${v_0}^2 < \frac{2\mu}{r_0}$, то e<1, траєкторія - еліпс.

якщо $h = 0$, тобто ${v_0}^2 = \frac{2\mu}{r_0}$, то e =1, траєкторія - парабола.

якщо $h > 0$, тобто ${v_0}^2 > \frac{2\mu}{r_0}$, то e >1, траєкторія - гіпербола.

Початкова швидкість, за якої точка може необмежено віддалитися від центра, має бути не менше за параболізну швидкість $v_n = \sqrt{\frac{2\mu}{r_0}}$. Як видно, швидкість v_n, що називається ще швидкістю звільнення, прямо пропорційна кореню квадратному з гаусівської сталої μ даного поля і обернено пропорційна кореню з початкової відстані r_0.

Щоб визначити закон руху точки вздовж її орбіти, скористаємося рівнянням (1.16) або (1.24): $\frac{d\varphi}{dt} = cu^2$.

Замінивши u його значенням із рівняння (1.35), отримаємо:

$$dt = u = \frac{p^2}{c}\frac{d\varphi}{(1 + e\cos\varphi)^2}$$

Звідки

$$t - t_0 = \frac{p^2}{c}\int_0^{\varphi}\frac{d\varphi}{(1+e\cos\varphi)^2}, \qquad (1.42)$$

де t_0 - момент проходження перицентра. Для обчислення інтегралу почнемо з підстановки:

$$\tan\frac{\varphi}{2} = \eta. \tag{1.43}$$

Тоді:

$$\cos\varphi = \frac{1-\eta^2}{1+\eta^2},\ d\varphi = \frac{2d\eta}{1+\eta^2} \tag{1.44}$$

Отримаємо після перетворень:

$$\frac{d\varphi}{(1+e\cos\varphi)^2} = \frac{2(1+\eta^2)d\eta}{(1+e)^2\left(1+\frac{1-e}{1+e}\eta^2\right)^2} \tag{1.45}$$

Розглянемо випадок, коли рух відбувається по еліптичній орбіті ($e < 1$) Візьмемо:

$$\sqrt{\frac{1-e}{1+e}} = k \text{ і } \vartheta = k\eta \tag{1.46}$$

Де нова змінна ϑ є в свою чергу половиною тангенсу деякого кута E. Звідси:

$$\vartheta = \tan\frac{\mathrm{E}}{2},\ d\vartheta = \left(\frac{1+\vartheta^2}{2}\right)dE,\ \cos E = \frac{1-\vartheta^2}{1+\vartheta^2}. \tag{1.47}$$

Тоді рівняння (1.45) набуде вигляду:

$$\frac{d\varphi}{(1+e\cos\varphi)^2}=\frac{2(k^2+\vartheta^2)d\vartheta}{k^3(1+e)^2(1+\vartheta^2)^2}=\frac{1-e+(1+e)\vartheta^2}{k^3(1+e)^3(1+\vartheta^2)}=\frac{1}{k^3(1+e)^3}\left(1-e\frac{1-\vartheta^2}{1+\vartheta^2}\right)dE=$$

$$\frac{1-e\cos E}{(1-e^2)^{3/2}}dE.$$

Підставляючи знайдене вираження в праву частину рівняння (1.42) і обчислюючи інтеграл, отримаємо остаточно:

$$E-e\sin E=\lambda(t-t_0), \tag{1.48}$$

де $\lambda=\frac{c(1-e^2)^{3/2}}{T}$. (1.49)

І, відповідно до (1.43), (1.46), (1.47):

$$\tan\frac{\mathrm{E}}{2}=\sqrt{\frac{1-e}{1+e}}\tan\frac{\varphi}{2}. \tag{1.50}$$

У небесній механіці кут φ називають істинною аномалією, а кут E - ексцентричною аномалією. Рівняння (1.48), що встановлює залежність між ексцентричною аномалією і часом, називається рівнянням Кеплера. Система рівнянь (1.35) і (1.48) дозволяє знайти r і φ в будь-який момент часу, тобто визначити закон руху точки в розглядуваному випадку.

Якщо позначити період обертання через T, то при $\varphi=\pi$ інтервал часу $(t-t_0)=\frac{T}{2}$, а $\tan\frac{\mathrm{E}}{2}=\infty$ і $E=\pi$. Відповідно, рівняння (1.48) дає $2\pi=\lambda T$. Звідси отримаємо вираження λ через період обертання:

$$\lambda=\frac{2\pi}{T}. \tag{1.51}$$

Таким чином, рівняння Кеплера можна ще представити у вигляді:

$$E - e\sin E = \frac{2\pi}{T}(t - t_0), \tag{1.52}$$

де, як уже зазначалося, t_0 - момент проходження перицентра.

При русі по параболічній орбіті ми, поклавши в рівнянні (1.45) $e = 1$ і обчисливши інтеграл (1.42), одразу отримаємо:

$$t - t_0 = \frac{1}{2}\tan\frac{\varphi}{2} + \frac{1}{6}\tan^3\frac{\varphi}{2}. \tag{1.53}$$

Розглянемо тепер випадок гіперболічної орбіти, $e > 1$. Із рівняння траєкторії (1.35) видно, що при зміні кута φ від нуля до значення φ^*, визначеного рівністю$e\cos\varphi^* = -1$, точка переміщується по відповідній гілці гіперболи від перицентра до нескінченності (кут φ^* задає напрямок асимптоти гіперболи).

Як видно з співвідношення (1.46), закон руху вздовж цієї гілки гіперболи можна знайти, поклавши в рівняннях (1.48) - (1.50) $E = iE_1$ і враховуючи, що $\sin iE_1 = i\sinh E_1$, $\tan iE_1 = i\tanh E_1$. Тоді для визначення закону руху отримаємо рівняння:

$$\mathrm{e}\sinh E_1 - E_1 = \lambda_1(t - t_0), \tag{1.54}$$

$$\text{де } \lambda_1 = \frac{c(e^2-1)^{3/2}}{T},\ \tan\frac{\mathrm{E}_1}{2} = \sqrt{\frac{e-1}{e+1}}\tan\frac{\varphi}{2}. \tag{1.55}$$

Останнє зі співвідношень (1.55) при зміні кута φ в інтервалі $0 \le \varphi \le \varphi^*$ завжди має сенс, так як якщо $e\cos\varphi \ge -1$, то $\sqrt{\frac{e-1}{e+1}}\tan\frac{\varphi}{2} \le 1$, в чому легко переконатися, виразивши $\cos\varphi$ через $\tan\frac{\varphi}{2}$.

У всіх попередніх розрахунках притягаючий центр (Сонце) вважався нерухомим відносно деякої інерціальної системи відліку (до зірок).

Уточнимо отримані результати, враховуючи взаємне притягання Сонця S і планети P, яка рухається навколо нього, і вважаючи відстань між цими тілами настільки великою порівняно з їхніми розмірами, що тіла можна розглядати як матеріальні точки.

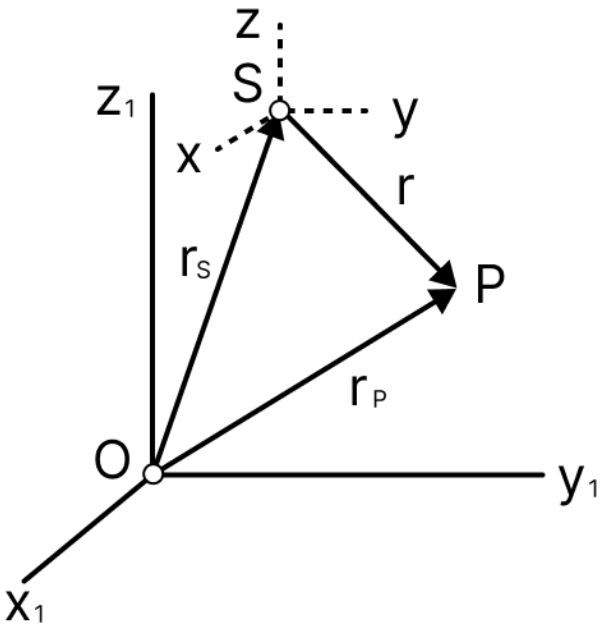

Рисунок 1.6 – Схематична візуалізація притягання Сонця та планети, яка рухається навколо нього

Нехай r_P - радіус-вектор планети P відносно деякої інерціальної системи відліку $Ox_1y_1z_1$, а r_S - радіус-вектор Сонця відносно тієї ж системи (рис. 1.6). Маси Сонця і планети позначимо відповідно M і m; тоді рівняння руху Сонця відносно системи відліку $Ox_1y_1z_1$ буде:

$$M\frac{d^2r_S}{dt^2} = \frac{fMm}{r^2}\frac{r}{r}, \qquad (1.56)$$

а рівняння руху планети відносно тієї ж системи:

$$m\frac{d^2r_P}{d\mathrm{t}^2} = \frac{fMm}{r^2}\frac{r}{r}, \tag{1.57}$$

де $r = \overline{SP}$ - радіус-вектор планети P відносно Сонця S, тобто відносно системи координат S_{xyz} , що переміщується разом з точкою S поступально відносно системи відліку $Ox_1y_1z_1$. Множачи обидві частини рівняння (1.56) на m, а рівняння (1.57) на M і віднімаючи почленно від рівняння (1.57) рівняння (1.56), отримаємо:

$$Mm\left(\frac{d^2r_P}{d\mathrm{t}^2} - \frac{d^2r_S}{d\mathrm{t}^2}\right) = -\frac{fMm}{r^2}\frac{r}{r}(M+m)$$

Звідси, так як $r_P - r_S = r$, отримаємо, скоротивши на M:

$$m\frac{d^2r}{d\mathrm{t}^2} = -\frac{fm(M+m)}{r^2}\frac{r}{r} = -\frac{\mu' m}{r^2}\frac{r}{r}; \tag{1.58}$$

де $\mu' = f(M+m)$. (1.59)

У рівняннях (1.56) і (1.57) при обчисленні похідних від r_S і r_P розглядається зміна цих векторів відносно осей $Ox_1y_1z_1$; отже, і в рівнянні (1.58) похідна теж береться по відношенню до цієї системи. Зазначимо, якщо осі S_{xyz} переміщуються поступально в системі $Ox_1y_1z_1$, то локальна похідна в осях S_{xyz} збігається з повною похідною в системі $Ox_1y_1z_1$.

Отже, рівняння (1.58) описує рух планети відносно системи координат, пов'язаної з Сонцем, тобто рух відносно Сонця. З цього рівняння видно, що рух планети навколо Сонця відбувається ніби навколо нерухомого тіла маси $M+m$, а не тільки M, як вважалося раніше.

У попередніх формулах цей результат легко врахувати, замінивши скрізь $\mu = fM$ на $\mu' = f(M+m)$.

Звідси випливає також, що гауссова стала поля тяжіння Сонця (або планети) фактично дорівнює не fM а $f(M+m)$, тобто вона залежить не тільки від маси притягаючого тіла, а й від маси тіла, яке рухається в полі тяжіння. Вважати $\mu = const$ можна лише як наближення в разі $M \gg m$.

Зазначимо, нарешті, що коли в полі тяжіння тіла S одночасно рухаються кілька тіл P_i (планет), то точне розв'язання задачі вимагає урахування не тільки притягання між тілами P_i і тілом S, а й взаємного тяжіння самих P_i. Точне розв’язання такої задачі, відоме як задача про рух n тіл, є математично надзвичайно складним [4]. До цього часу її не вдалося повністю вирішити за допомогою відомих в аналізі функцій навіть для випадку трьох тіл.

Виходячи з результатів, отриманих для задачі двох тіл, знайдемо відповідну поправку до третього закону Кеплера. Розглянемо рух навколо Сонця двох планет із масами m_1 і m_2. За формулами (1.29) і (1.59) будемо мати:

Для першої планети:

$$\frac{4\pi^2 a_1^3}{T_1^2} = \mu_1 = f(M+m_1)$$

Для другої планети:

$$\frac{4\pi^2 a_2^3}{T_2^2} = \mu_2 = f(M+m_2)$$

Поділивши перше рівняння на друге, отримаємо:

$$\left(\frac{a_1^3}{T_1^2}\right) : \left(\frac{a_2^3}{T_2^2}\right) = \frac{M+m_1}{M+m_2} = \frac{1+\frac{m_1}{M}}{1+\frac{m_2}{M}}, \qquad (1.60)$$

тоді як за третім законом Кеплера права частина рівняння (1.60) повинна дорівнювати одиниці. Отже, третій закон Кеплера має лише наближений характер і справедливий остільки, оскільки маси планет m_1 і m_2 малі порівняно з масою Сонця M.

Застосуємо отримані вище результати до вивчення руху тіла в полі тяжіння Землі. Вважатимемо Землю нерухомою, а тіло - матеріальною точкою. Опором повітря нехтуємо, що при великих висотах польоту є припустимим.

Нехай у початковий момент точка перебуває в положенні M_0 на відстані $R = OM_0$ від центру Землі (рис. 1.7), і нехай прискорення сили земного тяжіння в точці M_0 дорівнює g. Зазначимо, що під M_0 мається на увазі довільна висота над поверхнею Землі. Якщо ж точка M_0 береться на поверхні Землі, то $R = R_0$ - радіус земного екватора, $R_0 = 6378$ км, і $g = g_0 = 9{,}81$ м/с^2.

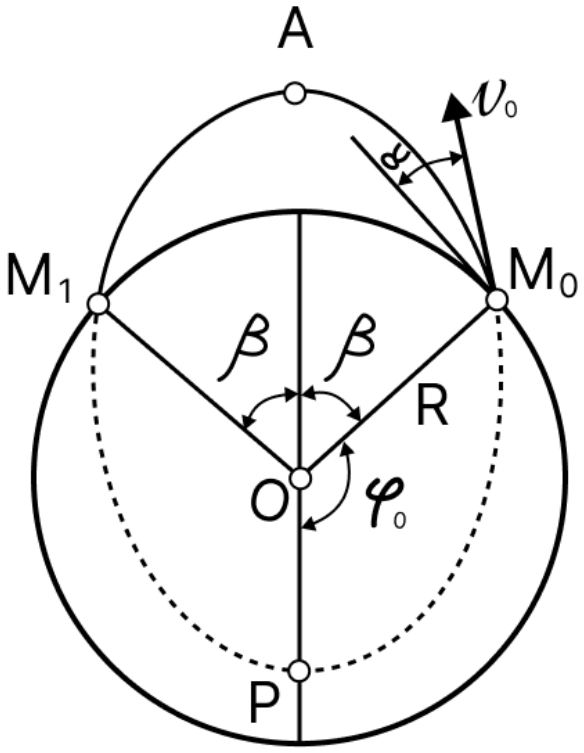

Рисунок 1.7 – Схема руху тіла в полі тяжіння Землі

Сила тяжіння, що діє на точку в положенні M_0, дорівнює $F_r - \frac{\mu m}{R^2} = -mg$.

Звідси отримаємо значення гауссової сталої поля земного тяжіння через R і g:

$$\mu = R^2 g. \tag{1.61}$$

Нехай у початковому положенні M_0 точка отримує початкову швидкість v_0, напрямлену під кутом α до горизонту. Тоді стала площі:

$$c = mom_0 v_0 = R v_0 \cos\alpha\,. \tag{1.62}$$

Траєкторія точки, як уже встановлено, - конічний переріз:

$$u = \frac{1+e\cos\varphi}{p} \text{ або } r = \frac{p}{1+e\cos\varphi}\,, \tag{1.63}$$

де кут φ відлічується від перигею P. Параметр p та ексцентриситет e визначаються за формулами (1.33) і (1.40), а значення кута $\varphi_0 = POM_0$ (див. рис. 1.7) можна знайти з формул (1.38).

Підставляючи скрізь c і μ з (1.61) і (1.62), і враховуючи $u_0 = \frac{1}{R}$, отримаємо:

$$p = \frac{c^2}{\mu^2} = \frac{{v_0}^2 \cos^2\alpha}{g}, \tag{1.64}$$

$$e = \sqrt{1 + \frac{{v_0}^2 \cos^2\alpha}{g^2 R^2}({v_0}^2 - 2gR)}, \tag{1.65}$$

Також:

$$e\sin\varphi_0 = \frac{{v_0}^2\cos^2\alpha}{gR}\tan\alpha,\ e\cos\varphi_0 = \frac{{v_0}^2\cos^2\alpha}{gR} - 1. \qquad (1.66)$$

З формули (1.65) видно, що траєкторія буде:

- еліпсом ($e < 1$) при $v_0 < \sqrt{2gR}$,
- параболою ($e = 1$) при $v_0 = \sqrt{2gR}$,
- гіперболою ($e > 1$) при $v_0 > \sqrt{2gR}$.

Параболічна швидкість (швидкість звільнення) $v_n = \sqrt{2gR}$ - це мінімальна початкова швидкість, за якої тіло зможе покинути поле тяжіння Землі. Також називається другою космічною швидкістю.

Якщо початкове положення взято на поверхні Землі, тобто $R = R_0 = 6378$ км $g = g_0 = 9{,}81$ м/с^2, то $v_n \approx 11{,}2$ км/с.

Тіло, що отримало початкову швидкість $v_0 \geq v_n$, напрямлену під великим кутом до горизонту, безперервно віддалятиметься від Землі, рухаючись по параболі або гіперболі (при $\alpha = 90°$ - по прямій).

При початковій швидкості $v_0 < v_n$, тіло або стане штучним супутником Землі, або впаде назад на Землю. Розглянемо ці випадки.

1) Штучні супутники. Для того, щоб тіло, кинуте з поверхні Землі, описало навколо Землі замкнуту криву, має бути у будь-якій точці траєкторії $r \geq R_0$ або, оскільки $R_0 = r(\varphi_0)$, то:

$$\frac{p}{1 + e\cos\varphi} \geq \frac{p}{1 + e\cos\varphi_0}$$

Отже, при зміні кута φ від 0 до π повинно бути $\varphi \geq \varphi_0$, що можливе лише при $\varphi_0 = 0$. Таким чином, перигей штучного супутника збігається з його початковим положенням M_0. Підставляючи значення $R = R_0$ та $\varphi_0 = 0$ у рівняння (1.66), отримаємо:

$$\cos^2\alpha \cdot \tan\alpha = 0, \frac{{v_0}^2 \cos^2\alpha}{g_0 R_0} - 1 = e. \quad (1.67)$$

Оскільки $e > 0$, то $\cos^2\alpha \neq 0$; отже, $\tan\alpha = 0$ і $\alpha = 0$ (або $\alpha = \pi$). Одночасно з останнього рівняння, враховуючи, що $\alpha = 0$, $e > 0$, отримуємо ${v_0}^2 > g_0 R_0$. Отже, щоб тіло, кинуте з поверхні Землі, перетворилося на штучний супутник Землі, необхідно виконати дві умови:

$$\alpha = 0\ , \sqrt{2g_0 R} > v_0 \geq \sqrt{g_0 R_0}. \quad (1.68)$$

Ексцентриситет орбіти супутника при будь-якому R, як видно з (1.67), буде:

$$e = \frac{{v_0}^2}{gR} - 1\ . \quad (1.69)$$

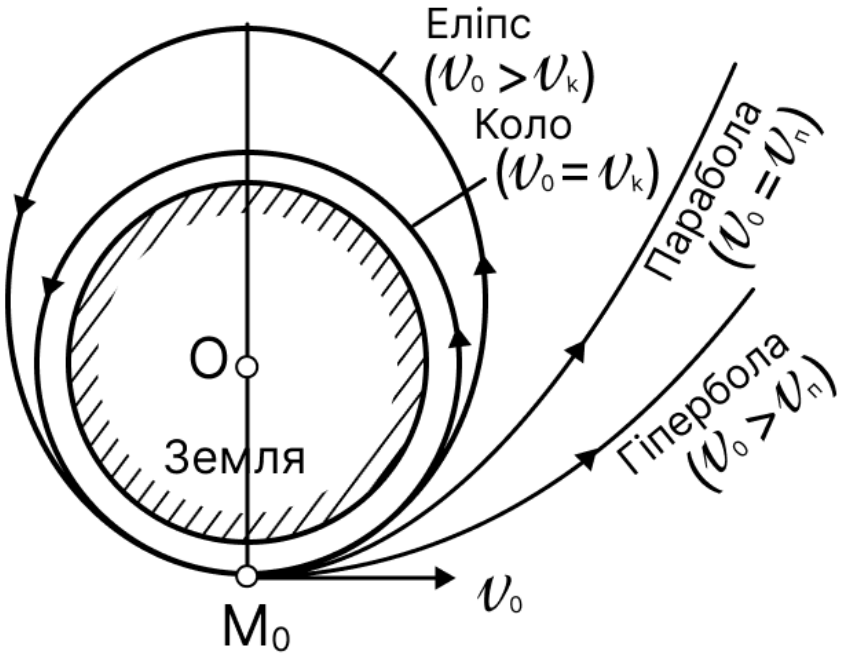

Рисунок 1.8 - Схематичне зображення кругової та еліптичної орбіт супутника Землі

Швидкість $v_k = \sqrt{gR}$, при якій $e = 0$ і супутник рухається по круговій орбіті радіуса R, називається круговою або першою космічною швидкістю; при киданні з поверхні Землі:

$v_k \geq \sqrt{g_0 R_0} \approx 7910$ м/сЯкщо $v_0 \geq v_k$, то орбіта супутника буде еліпсом, ексцентриситет якого тим більший, чим більший v_0 (рис. 1.8).

Коли кут кидання $\alpha \neq 0$, то при жодній початковій швидкості v_0 тіло, кинуто з поверхні Землі, не може стати супутником. На практиці для запуску штучного супутника використовують керовану ракету, яка підіймає супутник на задану висоту та надає йому у точці M_0 (рис. 1.8) необхідну швидкість v_0 під кутом $\alpha \approx 0$ до горизонту. Із збільшенням висоти H точки M_0 над поверхнею Землі стає можливими відхилення від умови $\alpha = 0$. Крім того, оскільки, за рівнянням (1.61):

$g = g_0 \frac{R_0^2}{R^2}$,

то при $R = R_0 + H$:

$$v_k = \sqrt{gR} = \sqrt{\frac{g_0 R_0^2}{R}} = \sqrt{g_0 R_0}\sqrt{\frac{R_0}{R_0 + H}}. \qquad (1.70)$$

Звідси випливає, що з ростом H величина кругової швидкості зменшується.

Період обертання T супутника можна знайти з третього закону Кеплера, виразивши його з рівняння (1.29). Замінивши в (1.29) гауссову сталу μ її значенням (1.61), маємо:

$$T = \frac{2\pi}{R}\sqrt{\frac{a^3}{g}}. \qquad (1.71)$$

Вхідну сюди велику піввісь орбіти можна, в свою чергу, виразити як:

$$a = \frac{r_P + r_A}{2} \text{ або } a = \frac{p}{1-e^2}, \tag{1.72}$$

Де r_P, r_A - відстані від центру Землі до перигею та апогею орбіти, а p, e - величини, визначені рівняннями (1.64) і (1.65); при $a = 0$ значення e визначається рівністю (1.69).

2) Еліптичні траєкторії. При $a > 0$ і $v_0 < \sqrt{2g_0R_0}$, тіло, кинуто з поверхні Землі, описавши дугу еліпса, впаде назад на Землю. Такі еліптичні траєкторії описують траєкторії снарядів і ракет. Знайдемо основні характеристики цих траєкторій.

Нехай D - горизонтальна дальність польоту. Позначивши центральний кут M_0OM_1 (див. рис. 1.7), отримаємо, що дальність вимірювання вздовж поверхні Землі:

$$D = \widetilde{M_0M_1} = 2R_0\beta \tag{1.73}$$

Оскільки $\beta = \pi - \varphi_0$ і $\tan\beta = -\tan\varphi_0$, то, ділячи рівняння (1.68) одне на одне, отримаємо:

$$\tan\beta = \frac{{v_0}^2\cos^2\alpha}{g_0R_0 - {v_0}^2\cos^2\alpha}\tan\alpha. \tag{1.74}$$

Формули (1.73), (1.74) і визначають величину дальності польоту за заданими значеннями v_0 і α. Зауважимо, що при $v_0\cos\alpha = \sqrt{g_0R_0}$, кут $\beta = \frac{\pi}{2}$, а дальність $D = \pi R_0$, тобто падіння буде у точці M_1, діаметрально протилежній початковому положенню M_0 (рис. 1.9).

Питання про найменшу початкову швидкість, необхідну для отримання заданої дальності, буде розглянуто нижче.

Найбільшу висоту траєкторії H можна знайти з рівняння:

$$H = r(\pi) - R_0$$

Покладаючи в рівнянні (1.63) $\varphi = \pi$ та замінюючи p його значенням з (1.64), отримаємо:

$$H = \frac{{v_0}^2 \cos^2 \alpha}{g_0(1-e)} - R_0. \qquad (1.75)$$

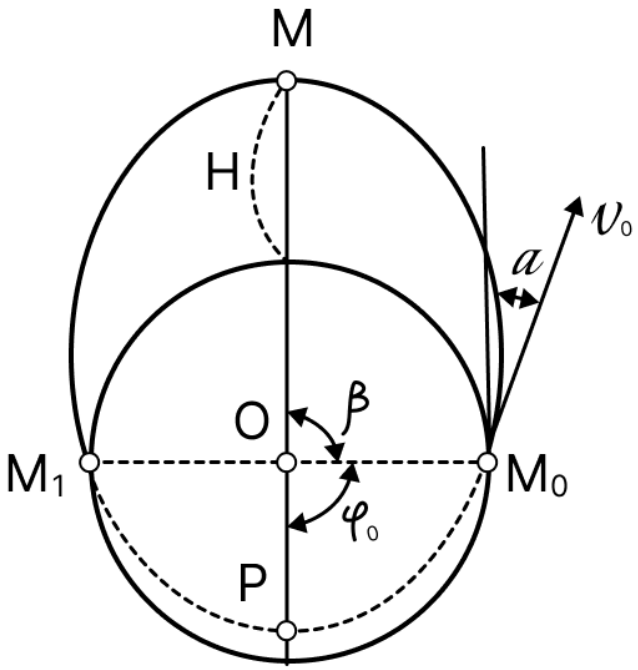

Рисунок 1.9 – Схема еліптичної траєкторії кинутого тіла з поверхні Землі

Час польоту з M_0 до M_1 (див. рис. 1.9) знайдемо за допомогою рівняння Кеплера (1.48), покладаючи в ньому для зручності $t_0 = 0$. Позначаючи період обертання по відповідній еліптичній орбіті як T, отримаємо:

$$T_{\text{польоту}} = 2[t(\pi) - t(\pi - \beta)] = 2\left[\frac{T}{2} - t(\pi - \beta)\right]$$

або, використовуючи рівняння (1.51) і (1.48):

$$T_{\text{польоту}} = \frac{2}{\lambda}(\pi - E_1 + e\sin E_1). \tag{1.76}$$

де E_1 - значення E при $\varphi = \pi - \beta$, а

$$\lambda = \frac{R_0 {g_0}^2 (1-e^2)^{3/2}}{{v_0}^3 \cos^3 \alpha} \tag{1.77}$$

Останній результат випливає з формул (1.49), (1.62) та (1.64), якщо в точці M_0 вважати $R = R_0$ та $g = g_0$.

Покладаючи в (1.76) $\pi - E_1 = E_\beta$, остаточно маємо:

$$T_{\text{польоту}} = \frac{2{v_0}^3 \cos^3 \alpha}{R_0 {g_0}^2 (1-e^2)^{3/2}}(E_\beta + e\sin E_\beta). \tag{1.78}$$

Відповідно до рівняння (1.50), де слід покласти $\varphi = \pi - \beta$, а $E = E_1 = \pi - E_\beta$, маємо:

$$\cot\frac{\mathrm{E}_\beta}{2} = \sqrt{\frac{1-e}{1+e}}\cot\frac{\beta}{2},\ \tan\frac{\mathrm{E}_\beta}{2} = \sqrt{\frac{1+e}{1-e}}\tan\frac{\beta}{2}, \tag{1.79}$$

а значення e дається рівністю (1.65).

Знайдемо тепер мінімальну початкову швидкість ${v_0}^{min}$ та відповідний їй найвигідніший кут кидання $a_{\text{п}}$, при якому можна досягти заданої дальності $D = 2R_0\beta$.

Для цього, виразивши початкову швидкість v_0 через β з рівняння (1.74), отримаємо:

$$v_0{}^2 = \frac{2g_0R_0 \tan\beta}{\sin 2\alpha + 2\cos^2\alpha \tan\beta}. \quad (1.80)$$

При даній дальності, тобто при даному куті β, величина v_0 залежить від α і буде найменшою, коли знаменник $F(\alpha) = \sin 2\alpha + 2\cos^2\alpha \tan\beta$ досягає максимуму. З рівняння $F'(\alpha) = 2\cos 2\alpha + 2\sin 2\alpha \tan\beta = 0$ Знаходимо $\cot 2\alpha = \tan\beta$, звідки

$$a_{\text{п}} = 45° - \frac{\beta}{2}. \quad (1.81)$$

Легко перевірити, що $F''(a_{\text{п}}) < 0$, отже при $\alpha = a_{\text{п}}$ маємо $v_0 = v_0^{min}$. Підставляючи значення $a_{\text{п}}$ у рівняння (1.80), отримаємо:

$$v_0^{min} = \sqrt{2g_0R_0 \frac{\sin\beta}{1+\sin\beta}}. \quad (1.82)$$

Формули (1.82) і (1.81) визначають мінімальну початкову швидкість, необхідну для досягнення заданої дальності $D = 2R_0\beta$, і кут кидання $a_{\text{п}}$, під яким ця швидкість має бути напрямлена до горизонту. Найвигідніший кут $a_{\text{п}}$ залежить від дальності і з її збільшенням зменшується; якщо дальність дуже мала, то $a_{\text{п}} \approx 45°$, як і в однорідному полі тяжіння.

Звернемо увагу на наступний результат. Представимо рівняння (1.80) у вигляді:

$$v_0{}^2 = \frac{2g_0R_0 \sin\beta}{2\cos\alpha \sin(\alpha+\beta)}. \quad (1.83)$$

Покладемо $\alpha' = 90 - (\alpha + \beta)$. Тоді буде $\cos\alpha' = \sin(\alpha + \beta)$, а $\sin(\alpha' + \beta) = \cos\alpha$ і ми отримаємо, що при даній дальності значення v_0 має таке ж значення при куті кидання α', як і при куті α. Отже, якщо при заданій початковій швидкості v_0 точка потрапляє в задану точку M_1, то це можливо за двома траєкторіями:

1) з кутом кидання $\alpha < a_{\text{п}}$ - настильна траєкторія;

2) з кутом $\alpha' = 90° - (\alpha + \beta) > a_{\text{п}}$- навісна траєкторія.

Якщо $\alpha = a_{\text{п}}$, то, як випливає з (1.81), і $\alpha' = a_{\text{п}}$, в цьому випадку обидві траєкторії зливаються в одну - найвигіднішу. Це властивість еліптичної траєкторії аналогічна властивості параболічної траєкторії у випадку однорідного поля тяжіння.

Отже, якщо величину кута $\beta \to 0$, тобто дальність $2R_0\beta = D$ розглядати в межі як малу горизонтальну дальність, то формули теорії еліптичної траєкторії переходять у відповідні формули для траєкторій параболічного кидання.

2 КЕРУВАННЯ РУХОМ ЛІТАЛЬНОГО АПАРАТУ

2.1 Математична модель руху

Розвиток науково-технічного прогресу в XXI столітті призвів до широкого впровадження безпілотних літальних апаратів (БПЛА) у різні сфери діяльності людини. Безпілотні системи активно використовуються як у військовій, так і в цивільній галузях: для відеоспостереження, моніторингу сільськогосподарських угідь, у сфері логістики, а також для розваг. З огляду на потребу в підготовці операторів та розробників БПЛА, особливого значення набуває створення симуляційних програмних середовищ, які дозволяють досліджувати системи управління, відпрацьовувати навички керування, а також реалізовувати аварійні сценарії, як-от автоматичне повернення до стартової точки.

Безпілотний літальний апарат (БПЛА) - це повітряний засіб, який здатен здійснювати політ без безпосередньої присутності пілота на борту. БПЛА вважається тільки той апарат, який знаходиться під постійним дистанційним контролем пілотів. Основними критеріями класифікації БПЛА є спосіб управління (автоматичне, дистанційне або некероване), тип двигуна, призначення, розміри та вага. Особливе місце серед БПЛА займають мультикоптери, зокрема квадракоптери - апарати з чотирма гвинтами, розташованими у горизонтальній площині.

Принцип польоту мультикоптерів базується на створенні тягових зусиль за допомогою обертання несучих гвинтів. Рівновага досягається завдяки симетричному розташуванню гвинтів та змінам їх швидкості обертання. Мультикоптери спроможні виконувати вертикальний зліт та посадку, повороти навколо своєї осі, переміщення у чотирьох напрямках, а також мають змогу стабільно зависати у повітрі. Різні типи маневрів (газ, крен, рискання) реалізуються за допомогою відповідних змін у швидкості обертання окремих пар гвинтів. Напрями руху гвинтів: ПЛГ(Г) – передній лівий гвинт (за год. стрілкою), ППГ(П) – передній правий гвинт (проти год.

стрілки), ЗЛГ(П) – задній лівий гвинт (проти год. стрілки), ЗПГ(Г) – задній правий гвинт (за год. стрілкою). (рис.2.1)

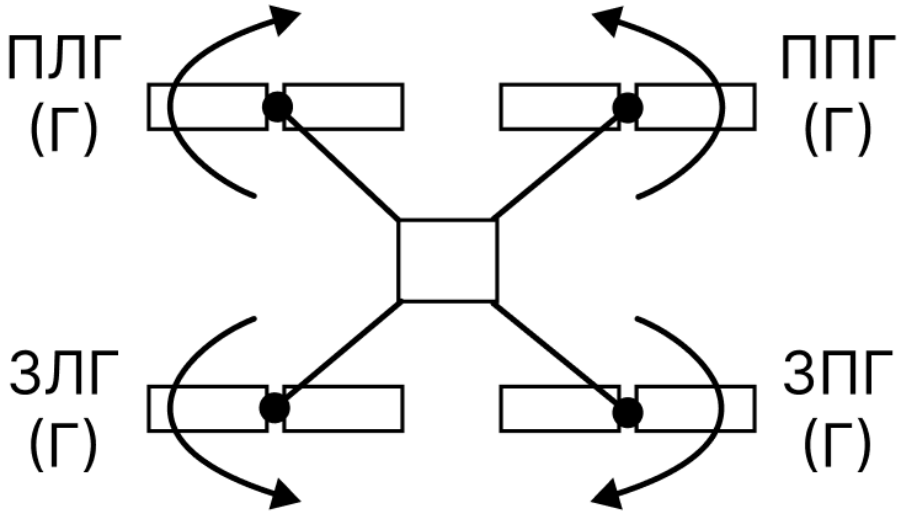

Рисунок 2.1 - Напрями руху гвинтів

Авіаційні симулятори - це програмні та апаратні комплекси, що імітують політ повітряного судна з метою підготовки операторів та випробування сценаріїв керування. Залежно від призначення симулятори поділяють на аркадні, реалістичні та професійні. Професійні симулятори використовуються у навчанні пілотів та включають складні апаратно-програмні компоненти, включаючи системи візуалізації, динамічну модель руху, панелі керування.

У статті “Дослідження системи управління безпілотних літальних апаратів” [5] автори створили симулятор квадрокоптера, що дозволяє користувачеві управляти віртуальним дроном через геймпад Logitech F510. Як зазначено: "Програма симуляції працює на ПК з ОС Windows, керування віртуальним засобом здійснюється за рахунок підключеного через порт USB контролеру управління". Основним середовищем розробки виступає Unity,

що дозволяє реалізовувати 3D-моделі та фізичні моделі у зручному інтерфейсі.

Розглянемо математичну модель квадрокоптера. Фізична модель квадрокоптера описується диференціальними рівняннями, які враховують сили тяжіння, сили тяги від гвинтів, кути орієнтації (тангаж, крен, рискання) [6]. Положення в просторі описується вектором координат $rT = (x, y, z)$, а орієнтація - вектором $\Omega T = (\psi, \theta, \varphi)$. Сила, створювана кожним гвинтом, визначається як

$F_i = b\omega_i^2$, де ω_i - кутова швидкість обертання двигуна, b - коефіцієнт пропорційності.

Це дозволяє створити модель, яка враховує зміну швидкості обертання, силу тяги, момент інерції, гравітацію. Врахування таких компонентів у симуляторі дозволяє досягти більшої фізичної реалістичності та прогнозованої поведінки віртуального апарату. (рис.2.2)

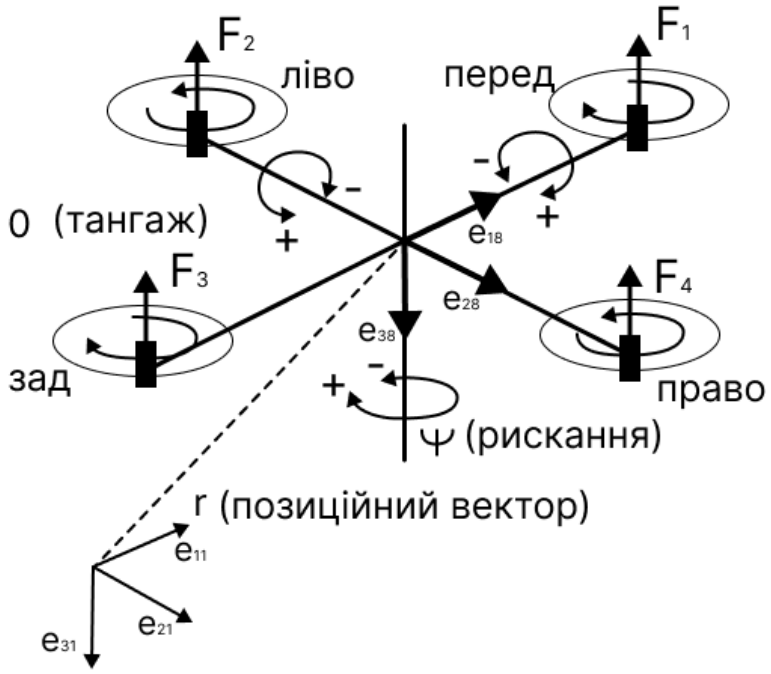

Рисунок 2.2 - Схема кутів, що визначають орієнтацію квадрокоптера у просторі

Надалі розглянемо алгоритм автоматичного повернення (А)*. Одним із важливих функціоналів симулятора є реалізація автоматичного повернення дрона у початкову точку при виникненні позаштатної ситуації. Для цього використано евристичний алгоритм А*, який належить до найефективніших методів пошуку оптимального шляху на графі. Пріоритет шляху визначається за значенням $f(x) = g(x) + h(x)''$, де $g(x)$ - вартість шляху до поточного вузла, $h(x)$ - евристичне наближення вартості до цільової вершини.

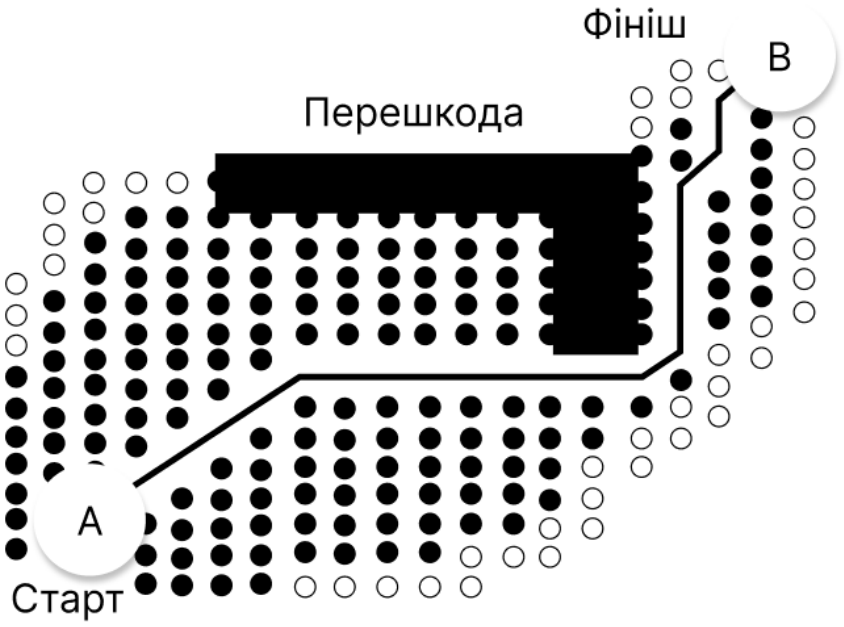

Рисунок 2.3 - Візуалізація реалізованого алгоритму А*

Процес реалізації включає розбиття карти на сітку, визначення прохідних та непрохідних клітин, оцінювання шляху до цілі, пошук обхідного маршруту у разі виявлення перешкод. Програма починає рух прямими лініями через необхідну послідовність вузлів. (рис.2.3) Таким чином, реалізується ефективна та надійна стратегія автономного повернення дрона. (рис.2.4)

Рисунок 2.4 - Блок-схема повернення БПЛА до стартової точки

Подібну програму можна реалізувати на базі Unity з використанням C#-скриптів. Для зв’язку з контролером Logitech використати API для зчитування осей стіків. Відповідність рухів осей і команд квадрокоптера виглядає так: ліва вертикальна вісь - газ (вгору-вниз), ліва горизонтальна - рискання (поворот по осі), права вертикальна - тангаж (вперед-назад), права горизонтальна - крен (вліво-вправо).

Фізична модель у середовищі Unity дозволяє обробляти зіткнення з об’єктами, зміну висоти, падіння при втраті керування. Квадрокоптер втрачає управління та може втрачати контроль при сильному зіткненні із землею або деревом.

Проведені дослідження підтверджують ефективність використання симуляторів для тестування систем керування БПЛА. Розроблена модель квадрокоптера дозволяє реалізовувати як ручне, так і автоматичне управління, враховуючи фізичні закономірності та алгоритмічну навігацію.

2.2 Математична модель керування вектором швидкості літального апарату

Математична модель керування вектором швидкості літального апарату (рис.2.5) призначена для поступового переналаштування поточного вектора швидкості $\overrightarrow{V_0}$ апарату на цільовий вектор $\overrightarrow{V_k}$, з урахуванням обмежень на зміну модуля вектора та напрямку (через кутові косинуси). Такий підхід дозволяє моделювати плавне, стабільне управління орієнтацією і швидкістю в умовах, наближених до реальних – наприклад, в динамічних аерокосмічних або безпілотних системах.

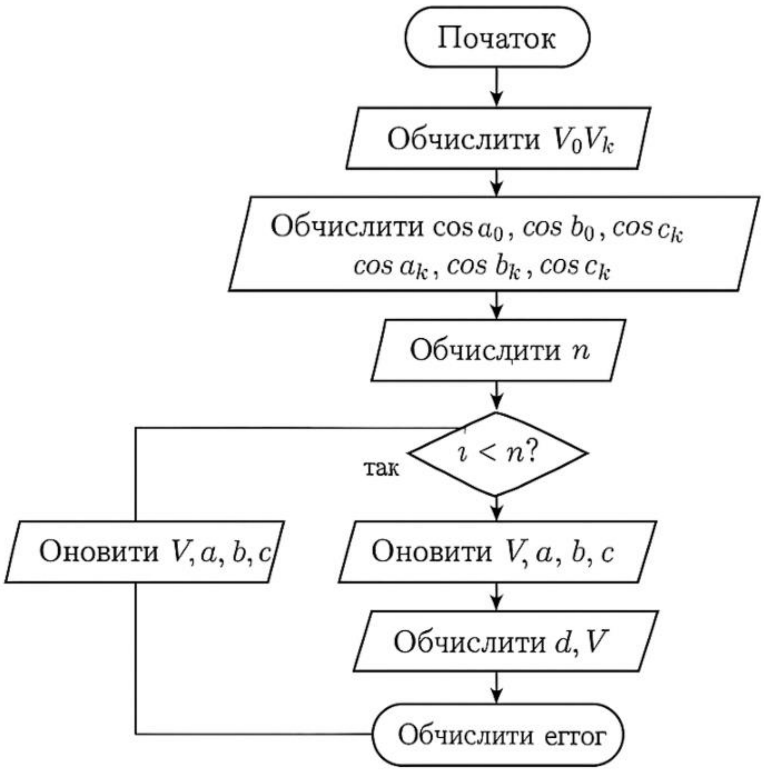

Рисунок 2.5 - Блок-схема математичної моделі керування вектором швидкості літального апарату

Для початку визначаємо два тривимірні вектори: $\overrightarrow{V_0} = (V_{0x}, V_{0y}, V_{0z})$ та $\overrightarrow{V_k} = (V_{kx}, V_{ky}, V_{kz})$.

Ці вектори відповідно описують поточну та цільову швидкість літального апарату в просторі. Щоб перейти від одного до іншого, обчислюємо їх модулі: $|\overrightarrow{V_0}| = \sqrt{V_{0x}^2 + V_{0y}^2 + V_{0z}^2}$, $|\overrightarrow{V_k}| = \sqrt{V_{kx}^2 + V_{ky}^2 + V_{kz}^2}$.

Ці значення описують абсолютну величину швидкості в кожній точці.

Кожен вектор швидкості можна представити як добуток модуля вектора та одиничного напрямного вектора: $\vec{V} = V \cdot \vec{d}$, $\vec{d} = (\cos\alpha, \cos\beta, \cos\gamma)$, де $\cos\alpha = \frac{V_x}{|\vec{V}|}$, $\cos\beta = \frac{V_y}{|\vec{V}|}$, $\cos\gamma = \frac{V_z}{|\vec{V}|}$. Ці значення - це напрямні косинуси, що є проєкціями одиничного напрямного вектора на координатні осі. Вони визначають орієнтацію руху апарату в тривимірному просторі.

Задача алгоритму - змінити вектор $\overrightarrow{V_0}$ на $\overrightarrow{V_k}$, змінюючи поступово як модуль швидкості, так і її напрямок. Для цього спочатку обчислюємо початкові та кінцеві значення кутів напрямку: $\alpha_0 = \arccos\left(\frac{V_{0x}}{|\overrightarrow{V_0}|}\right)$, $\alpha_k = \arccos\left(\frac{V_{kx}}{|\overrightarrow{V_k}|}\right)$, $\beta_0 = \arccos\left(\frac{V_{0y}}{|\overrightarrow{V_0}|}\right)$, $\beta_k = \arccos\left(\frac{V_{ky}}{|\overrightarrow{V_k}|}\right)$, $\gamma_0 = \arccos\left(\frac{V_{0z}}{|\overrightarrow{V_0}|}\right)$, $\alpha_z = \arccos\left(\frac{V_{kz}}{|\overrightarrow{V_k}|}\right)$.

Таким чином, для кожного з трьох напрямків ми маємо початковий і кінцевий кут, який потрібно подолати поступово, щоб змінити орієнтацію вектора швидкості.

На цьому етапі визначається, скільки кроків потрібно для поступового переходу від $\overrightarrow{V_0}$ до $\overrightarrow{V_k}$. Для цього розраховуються кількості змін окремо для модуля та кожного з кутів: $n_V = \left\lceil \frac{|V_k - V_0|}{\Delta V} \right\rceil$, $n_\alpha = \left\lceil \frac{|\alpha_k - \alpha_0|}{\Delta\alpha} \right\rceil$. Аналогічно визначається кількість кроків для β та γ: $n_\beta \left\lceil \frac{|\beta_k - \beta_0|}{\Delta\beta} \right\rceil$, $n_\gamma \left\lceil \frac{|\gamma_k - \gamma_0|}{\Delta\gamma} \right\rceil$, після чого загальна кількість ітерацій визначається як: $n = max(n_V, n_\alpha, n_\beta, n_\gamma)$

Цей механізм забезпечує адаптивність: якщо змінити один з параметрів різко, то відповідно збільшується кількість кроків, а значить - плавність та точність керування.

В основному циклі алгоритму на кожному кроці виконується обчислення приростів значень: $step_V = \frac{V_k - V}{n-i+1}$, $step_\alpha = \frac{\alpha_k - \alpha}{n-i+1}$, $step_\beta = \frac{\beta_k - \beta}{n-i+1}$, $step_\gamma = \frac{\gamma_k - \gamma}{n-i+1}$.

Це гарантує зменшення розміру кроку при наближенні до цільового значення. Далі оновлюються значення: $V := V + step_V$, $\alpha := \alpha + step_\alpha$, $\beta := \beta + step_\beta$, $\gamma := \gamma + step_\gamma$.

Нові кутові значення переводяться в косинуси: $\cos\alpha = \cos(\alpha)$, $\cos\beta = \cos(\beta)$, $\cos\gamma = \cos(\gamma)$.

Потім ці значення нормалізуються для створення одиничного напрямного вектора: $\vec{d} = \frac{(\cos\alpha, \cos\beta, \cos\gamma)}{\sqrt{\cos^2\alpha + \cos^2\beta + \cos^2\gamma}}$

Поточний вектор швидкості на цьому кроці обчислюється як: $\vec{V}_{curr} = V \cdot \vec{d}$.

Після завершення всіх ітерацій визначається відстань між останнім обчисленим вектором та цільовим вектором швидкості: $error = \left|\vec{V}_{curr} - \vec{V}_k\right| = \sqrt{\left(V_{x,curr} - V_{kx}\right)^2 + \left(V_{y,curr} - V_{ky}\right)^2 + \left(V_{z,curr} - V_{kz}\right)^2}$.

Це значення характеризує точність, з якою модель досягла бажаного вектора. Воно може використовуватись як критерій завершення або оцінки ефективності алгоритму. Якщо похибка перевищує допустиме значення - модель можна повторити з меншими кроками або уточненими параметрами. Таким чином, обчислення похибки є ключовим етапом для перевірки якості управління та адаптації моделі в майбутніх циклах керування.

Логіка роботи наведеної математичної моделі ґрунтується на динамічному керуванні вектором швидкості літального апарату з урахуванням як його модуля, так і просторового напрямку. Алгоритм

побудований таким чином, що на кожному кроці розраховується залишкова відстань між поточними та цільовими параметрами - це дозволяє адаптивно визначати величину змін у швидкості та напрямку. Завдяки цьому досягається плавність керування, що є критично важливим при роботі з реальними аеродинамічними об'єктами, де різкі зміни можуть призводити до дестабілізації або втрати керованості.

Модель є універсальною, оскільки працює з будь-якими просторовими напрямками вектора швидкості. Вона не залежить від конкретного положення об'єкта у просторі або особливостей координат - обчислення базуються на кутових косинусах, що забезпечує незалежність від обраної системи відліку. Це дає змогу застосовувати модель як для горизонтального, так і для вертикального чи складного тривимірного маневрування.

Одним із ключових компонентів моделі є нормалізація напрямного вектора. Після кожного обчислення напрямку новий вектор приводиться до одиничної довжини, що гарантує фізичну коректність і стабільність результату. У контексті реальних польотів це означає, що орієнтація апарату завжди буде відповідати дійсному напрямку руху, без викривлень або відхилень, які можуть бути спричинені накопиченням похибок.

Точність досягається за рахунок адаптивного механізму перерахунку кількості ітерацій (кроків) у процесі руху до цілі. Замість фіксованої кількості змін, алгоритм автоматично реагує на зміну відстані до цільового стану, забезпечуючи оптимальну кількість кроків у поточному контексті. Це особливо корисно в умовах, де довжина вектора чи його орієнтація змінюється нерівномірно, а також коли важливо підтримувати баланс між швидкістю зближення і обчислювальними ресурсами.

Гнучкість цієї математичної моделі полягає у тому, що вона дозволяє змінювати вхідні параметри відповідно до конкретної задачі чи апаратних обмежень. Наприклад, параметр mmm дозволяє регулювати кількість основних контрольних кроків, а зміна ΔV, $\Delta\alpha$, $\Delta\beta, \Delta\gamma$ забезпечує налаштування чутливості моделі до зміни швидкості та орієнтації. Це

відкриває можливість використовувати алгоритм у широкому спектрі практичних задач.

Застосування моделі охоплює низку критично важливих галузей. Вона може бути використана у навігаційних системах безпілотних літальних апаратів, де точність керування напрямком руху є основою ефективного виконання місій. У системах автоматичного наведення літальних апаратів модель забезпечує обчислення траєкторії підходу до цілі з дотриманням обмежень по маневреності. Вона також актуальна для використання у тренажерах для навчання пілотів або операторів, де потрібна реалістична і передбачувана симуляція зміни швидкості та напрямку. Крім того, алгоритм придатний для вбудовування в системи корекції курсу ракет або супутників, де потрібно точне й стабільне регулювання вектора швидкості з урахуванням багатьох зовнішніх факторів.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТЕХНОЛОГІЇ ДОСЛІДЖЕННЯ

3.1 Платформа для розробки системи дослідження

У межах реалізації дослідницької частини дипломної роботи як базову платформу було обрано програмне середовище Blender [7]. Це сучасне, кросплатформне програмне забезпечення з відкритим вихідним кодом, яке поєднує можливості тривимірного моделювання, анімації, візуалізації, фізичної симуляції та обробки відео. Blender є визнаним інструментом у сфері комп’ютерної графіки, який постійно розвивається і підтримується потужною спільнотою користувачів та розробників. Його популярність обумовлена поєднанням професійної функціональності, повної безкоштовності та гнучкої архітектури з можливістю інтеграції зовнішніх модулів і сценаріїв.

Окремої уваги заслуговує підтримка мови програмування Python, яка відіграє ключову роль у розробці та реалізації прикладних алгоритмів у межах цієї роботи. Завдяки інтеграції Python у середовище Blender [8], користувач отримує доступ до внутрішнього API програми через модуль bpy, що дає змогу програмно створювати та змінювати об’єкти сцени, обчислювати параметри руху, здійснювати рендеринг зображень, керувати анімацією та реалізовувати логіку взаємодії між об'єктами. Python використовується для реалізації математичної моделі керування вектором швидкості літального апарата, зокрема для обчислення кутових косинусів, нормалізації векторів, поступового переналаштування напрямку швидкості та реалізації умовних обмежень, що моделюють реальні фізичні властивості системи.

Blender знаходить широке застосування в численних галузях. У кіноіндустрії та сфері анімації його використовують для створення фотореалістичних тривимірних сцен, спецефектів, анімаційних фільмів та короткометражок. У галузі ігрової розробки Blender застосовується для створення 3D-моделей персонажів, середовища, механіки та анімацій, які

потім експортуються до рушіїв, таких як Unity або Unreal Engine [9]. В архітектурі платформа слугує засобом візуалізації інтер'єрів та екстер'єрів будівель, дозволяючи створювати інтерактивні презентації та віртуальні тури. У дизайні промислових виробів і техніки Blender дозволяє моделювати концептуальні прототипи об'єктів із високим рівнем деталізації та реалістичним освітленням. Візуальні ефекти, створені в Blender, активно використовуються у рекламі, віртуальній та доповненій реальності, медіа-продуктах та онлайн-освіті.

Окрім вищезазначених сфер, Blender набув особливого значення в академічних дослідженнях, інженерній діяльності та технічному моделюванні. У наукових візуалізаціях Blender використовується для побудови моделей молекул, реконструкції геологічних процесів, симуляцій руху небесних тіл, а також для створення симуляцій у фізиці та механіці. У галузі робототехніки Blender застосовується для імітації руху роботизованих механізмів, перевірки моделей зворотного зв'язку, візуалізації траєкторій та взаємодії з навколишнім середовищем. У медичній сфері за допомогою Blender створюються тривимірні реконструкції органів, симуляції хірургічних втручань, навчальні візуалізації для підготовки майбутніх лікарів. У аерокосмічній галузі платформа застосовується для моделювання літальних апаратів, дослідження аеродинамічних характеристик, побудови польотних траєкторій та перевірки систем навігації.

У контексті цього дослідження, можливості Blender були використані для побудови тривимірного представлення векторів швидкості об'єкта, а також для наочної демонстрації реалізації алгоритму керування з урахуванням фізичних обмежень. Використання Python [10], [11] забезпечило реалізацію покрокових обчислень усередині Blender без необхідності звертатися до сторонніх програмних засобів, що значно спростило інтеграцію логіки математичної моделі з графічним середовищем.

Інтерактивна візуалізація, яку забезпечує Blender, дозволяє здійснювати аналіз результатів не лише у вигляді числових даних, а й через

графічне представлення - з можливістю фіксації змін у просторі, візуального контролю напряму руху та оцінки відхилень між поточними й цільовими параметрами. Такий підхід має безсумнівну цінність у дослідженнях, де точність алгоритмічного керування поєднується з необхідністю візуального підтвердження його ефективності.

Таким чином, Blender у поєднанні з Python виступає не лише потужним інструментом для візуалізації, а й універсальною платформою для повноцінної реалізації та тестування дослідницьких алгоритмів. Його застосування дозволило забезпечити високу інтеграцію візуального середовища з програмною логікою та створити гнучку систему, придатну для подальшого розширення, адаптації та практичного використання в суміжних технічних і наукових галузях.

3.2 Декомпозиція системи

Для реалізації візуальної складової дослідження в середовищі Blender була створена повноцінна тривимірна сцена, що відображає поведінку літального апарата (дрона) під час зміни його вектора швидкості відповідно до розробленого алгоритму. Побудова сцени ґрунтується на поєднанні геометричних об'єктів, матеріалів, логічних структур управління, а також програмної анімації, яка відображає зміну параметрів руху в динаміці.

Центральним об'єктом сцени є тривимірна модель дрона, що була створена безпосередньо в Blender за допомогою стандартних примітивів. Модель має складену структуру: корпус сформовано з кубічних та сферичних об’єктів, які відображають основні функціональні частини дрона, зокрема центральний модуль, двигуни, пропелери та стабілізаційні елементи. Всі ці об’єкти об’єднані в одну логічну групу за допомогою ієрархії прив’язки до об’єкта Empty, який виконує роль базової точки керування. Саме до цього об’єкта застосовується анімація руху, яка у свою чергу передається всій 3D-моделі дрона через механізм наслідування трансформацій.

Кожен геометричний компонент дрона має призначений матеріал, створений у редакторі шейдерів Blender. Матеріали забезпечують візуальну виразність сцени, відтворюючи поверхневі властивості елементів дрона: металеві відблиски, матову текстуру пластику, прозорість оптичних компонентів тощо. Для підвищення реалістичності було налаштовано параметри освітлення матеріалів відповідно до типу поверхні.

У сцену додано освітлення, яке складається з одного джерела типу Area Light, розміщеного над моделлю дрона. Це забезпечує м’яке рівномірне освітлення, що підкреслює форму і текстуру моделі без сильних тіней. Додатково, у разі необхідності акцентування руху або векторів, можуть використовуватись точкові або направлені джерела світла, які розміщуються відповідно до положення камери.

Для фіксації сцени та подальшого рендерингу було також розміщено камеру, орієнтовану таким чином, щоб охоплювати весь простір переміщення дрона. Камера встановлена під перспективним кутом, з орієнтацією, що дозволяє спостерігати зміну положення та напрямку руху у трьох координатних вимірах. Її положення є фіксованим протягом усієї анімації, однак у разі потреби вона може бути анімована для супроводу об’єкта в русі.

Для візуального представлення векторів руху у сцену були додані об’єкти типу Plain Axes, які імітують напрямки швидкості. Зокрема, створено три векторні вказівники: StartVector (початковий напрямок), EndVector (цільовий напрямок) та SpeedVector (динамічний поточний напрямок). Їхнє положення, орієнтація та масштаб змінюються відповідно до даних, обчислених у Python-скрипті. Вектори масштабуються по осі Z пропорційно до величини швидкості, що дозволяє візуально розрізняти інтенсивність руху на різних етапах симуляції.

Програмна логіка реалізована мовою Python з використанням бібліотек numpy для математичних обчислень та mathutils (модуль Blender) для векторних операцій [12], [13]. У Python-скрипті реалізовано алгоритм

керування вектором швидкості, який розраховує поступове переналаштування з вектора V_0 на V_k шляхом зміни як модуля, так і орієнтації. Об’єкт Empty, до якого прив’язана модель дрона, переміщується згідно з обчисленими координатами. На кожному кадрі визначається новий напрямок вектора, масштабується відповідний візуальний елемент SpeedVector, а об'єкт дрона отримує нову позицію у просторі через ключові кадри (keyframe_insert), які записуються автоматично в межах циклу ітерацій.

Ключовою особливістю такої декомпозиції системи є чітке розмежування логічних, графічних та програмних компонентів. Модель дрона виконує роль візуалізації фізичного об’єкта; об’єкт Empty є опорною точкою анімації та руху; вектори Plain Axes - засобом відображення напрямку і швидкості; освітлення та камера забезпечують реалістичну фіксацію сцени для візуального аналізу; а Python-скрипт виступає ядром обчислювального механізму, що визначає всю поведінку системи.

Такий підхід до декомпозиції дозволяє не лише ефективно візуалізувати складні математичні моделі в реальному часі, але й забезпечує можливість гнучкої модифікації кожного компонента, що є критично важливим у процесі наукового дослідження та подальшого розширення функціоналу.

3.3 Модуль візуалізації руху системи тіл

У модулі (див. Додаток А) реалізовано програмну візуалізацію поступової зміни вектора швидкості у тривимірному просторі з використанням середовища Blender. Основна мета - відобразити процес переходу від початкового вектора швидкості до цільового з поступовим наближенням як за модулем, так і за напрямком, що відповідає умовам контрольованого керованого переходу між двома станами.

На першому етапі програма ініціалізує об’єкти сцени, зокрема об’єкт, який позначає поточну позицію вектора швидкості, об'єкт для візуалізації

самого вектора, а також два орієнтири, що позначають початковий і кінцевий вектори. Для кожного з них очищується попередня анімація.

Вектори початкового та кінцевого стану задаються як тривимірні масиви координат. Для обчислення кута між ними програма спочатку визначає їхні модулі, після чого розраховує косинуси напрямків по кожній осі. З цих значень визначаються відповідні кутові координати в сферичній системі. Це дає змогу працювати не лише з величиною швидкості, а й з її напрямом у просторі. На основі різниці між початковими і цільовими значеннями модуля та кутів визначається кількість ітерацій (кадрів) для поступового переходу, з урахуванням заданих максимальних приростів для кожного параметра.

Далі відбувається основний процес анімації. У циклі кожен крок відповідає одному ключовому кадру. На кожному з них програма коригує поточне значення модуля вектора швидкості, а також кутові координати, наближаючи їх до цільових. Обчислюються прирости - тобто кроки, якими величини мають змінюватись при збереженні плавності руху. Потім на основі оновлених кутів заново обчислюється напрямок вектора, нормалізується його орієнтація і множиться на поточну величину модуля швидкості. Таким чином, формується нове положення вектора швидкості у просторі.

Після цього положення основного об’єкта змінюється відповідно до нового вектора, вставляється ключовий кадр для збереження анімації, і візуалізується сам вектор швидкості - змінюється його положення, масштаб і орієнтація. Останнє виконується шляхом обчислення повороту від стандартного напрямку (уздовж осі Z) до напрямку поточного вектора. Це дозволяє візуально зорієнтувати стрілку вектора відповідно до фактичного напрямку руху.

На завершення перевіряється, наскільки близьким є обчислений вектор до цільового. Якщо залишкова похибка перевищує прийнятну, то програма вставляє фінальний кадр з точним розташуванням вектора швидкості у

цільовій позиції, забезпечуючи коректне завершення процесу. Крім того, програма розміщує початковий і кінцевий вектори у сцені як нерухомі орієнтири, відображаючи початковий і кінцевий напрямки та довжини відповідних векторів.

Таким чином, створено модуль, який реалізує покадрову зміну вектора швидкості в тривимірному просторі (рис 3.1), наочно демонструючи як зміну його модуля, так і просторової орієнтації, із забезпеченням коректного візуального супроводу на кожному етапі.

Рисунок 3.1 – Анімація руху вектора швидкості

У наступному модулі (див. Додаток Б) реалізовано візуалізацію просторового руху об’єкта у середовищі Blender на основі розв’язання системи диференціальних рівнянь, які описують політ у полі тяжіння. Рух здійснюється у вертикально-площинному напрямку (площина XZ), без урахування впливу повітряного опору. Об’єктом, що рухається, виступає умовна точка у вигляді порожнього об’єкта (Empty), траєкторія якого відповідає класичній параболі польоту.

Фізична модель задається у вигляді системи диференціальних рівнянь другого порядку. За другим законом Ньютона, вздовж осі X на тіло не діє

жодна сила, тому прискорення дорівнює нулю: $m\ddot{x} = 0$, що дає в результаті $\ddot{x} = 0$. Відповідно, рух вздовж осі X є рівномірним: $x(t) = v_0 \cos\alpha \cdot t$. Уздовж осі Z на тіло діє сила тяжіння: $m\ddot{z} = -mg$, звідки $\ddot{z} = -g$, тобто рух є рівноприскореним. Розв'язок має вигляд: $z(t) = z_0 + v_0 \sin\alpha \cdot t - \frac{1}{2} g t^2$. Початкові умови задаються явно: координата X на початку дорівнює нулю, початкова швидкість має модуль v_0 і кут α, початкова висота - z .

На основі цих рівнянь програма виконує покадрову генерацію положення об'єкта. Користувач задає початкову швидкість, кут вильоту у градусах, початкову висоту та значення прискорення вільного падіння. Кут переводиться у радіани, після чого обчислюються тригонометричні компоненти. Для того, щоб точно визначити тривалість руху до моменту, коли об'єкт торкнеться землі (тобто рівня $z = 0$), обчислюється аналітичний час польоту на основі розв'язання квадратного рівняння, яке утворюється із виразу для $z(t)$.

Цей час перетворюється у кількість кадрів відповідно до заданої частоти анімації. Основний цикл обчислення проходить по кожному кадру. Для кожного моменту часу розраховується положення об'єкта за координатами X і Z згідно з аналітичними рівняннями. Якщо значення Z стає меншим за нуль, що фізично означає приземлення, то значення примусово встановлюється на рівень нуля, і об'єкт отримує фінальне положення. У кожному кадрі положення об'єкта записується через вставку ключових кадрів, що забезпечує плавну анімацію в Blender.

У результаті реалізовано точне чисельне моделювання траєкторії тіла, що рухається у гравітаційному полі (рис 3.2). Візуалізація відповідає фізичним законам і дозволяє не лише спостерігати за польотом об'єкта, а й точно контролювати його параметри через змінні початкової швидкості, кута запуску та висоти. Модуль є універсальним прикладом застосування рівнянь механіки для анімації у графічному середовищі.

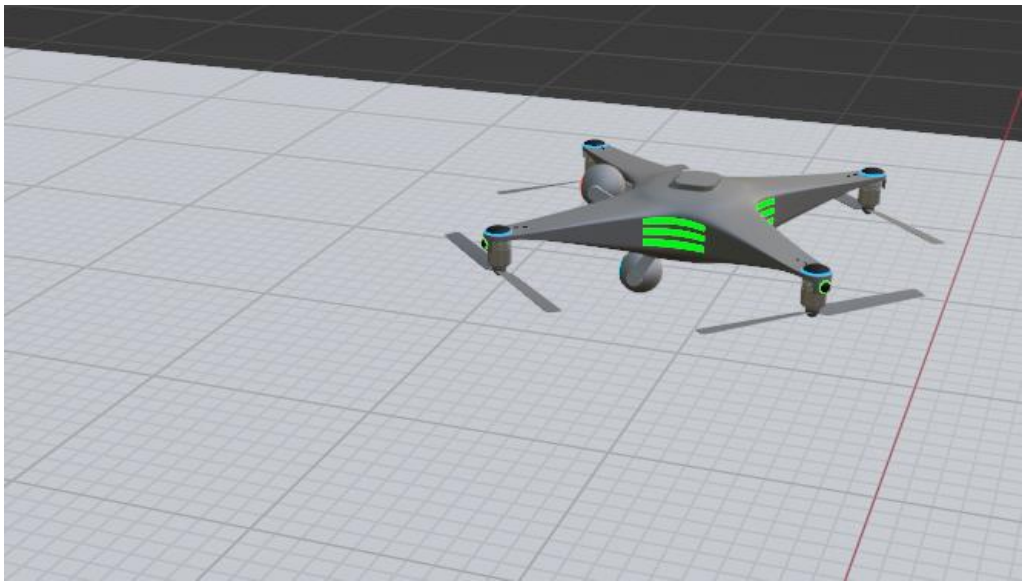

Рисунок 3.2 – Анімація руху дрону, заданого диференціальними рівняннями

ВИСНОВОК

У дипломній роботі розроблено інформаційну технологію для моделювання та керування рухом літального апарату в умовах змінного гравітаційного поля з візуалізацією у тривимірному середовищі.

Для реалізації поставленої мети були вирішені наступні задачі:

1. Проаналізовано фізичні особливості та математичні моделі руху тіл у гравітаційному полі.
2. Розроблено алгоритм керування рухом літального апарату шляхом поступового змінення вектора швидкості відповідно до цільового напрямку, з урахуванням обмежень на зміну модуля та напрямку.
3. Побудовано математичну модель для поступової корекції орієнтації та швидкості об’єкта.
4. Реалізовано чисельний алгоритм у середовищі програмування Python з використанням бібліотек NumPy та mathutils.
5. Створено віртуальну лабораторію у Blender для наочного моделювання траєкторії руху, анімації зміни положення об'єкта та векторів швидкості.
6. Проведено експериментальне дослідження поведінки системи при різних початкових умовах і параметрах, оцінено коректність роботи алгоритму.

Результатом виконання дипломної роботи є програмна реалізація інформаційної технології, яка дозволяє досліджувати поведінку літального апарату в умовах складного руху та візуально аналізувати динаміку зміни його швидкості та орієнтації у просторі. Отримані результати можуть бути використані в освітніх цілях, а також при розробці систем керування для безпілотних або автоматизованих літальних пристроїв.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Мельник О. Г. Теоретична механіка: підручник. - Львів: Видавництво Львівської політехніки, 2015. - 412 с.

2. Pasternak, B., & Sobczyk, K. (2008). Mathematics for Physics and Physicists. Cambridge University Press. - 544 p.

3. Ковальчук В. А. Чисельні методи в задачах механіки: навч. посіб. - Вінниця: ВНТУ, 2020. - 188 с.

4. Гаврилюк І. О., Мисак Ю. Б. Математичне моделювання фізичних процесів: навч. посіб. - Тернопіль: ТНТУ, 2018. - 210 с.

5. Подорожняк А. А. Дослідження системи керування об’єктом з використанням віртуального середовища. Автоматизація, інструменти та технології, 2018, №2(3), с. 43-48. - URL: https://ep3.nuwm.edu.ua/19439/

6. Лі М. Інженерна механіка в задачах: перекл. з англ. - Київ: Наукова думка, 2019. - 260 с

7. Blender Foundation. Blender Manual. - URL: https://docs.blender.org/manual/en/latest/

8. Blender Python API - Mathutils documentation. - URL: https://docs.blender.org/api/current/mathutils

9. McCarthy, P. (2020). Mastering Blender (4th ed.). Birmingham: Packt Publishing. - 528 p.

10. Python Software Foundation. Python 3.10 Documentation. - URL: https://docs.python.org/3.10/

11. VanderPlas, J. (2016). Python Data Science Handbook: Essential Tools for Working with Data. O’Reilly Media. - 548 p.

12. NumPy Developers. NumPy Documentation. - URL: https://numpy.org/doc.

13. Oliphant, T. E. (2015). Guide to NumPy (2nd ed.). USA: Continuum Analytics. - 320 p.

ДОДАТОК А. Код скрипту математичної моделі руху вектора швидкості

```
import bpy
import numpy as np
from mathutils import Vector

# Ініціалізація об'єктів
drone = bpy.data.objects["Empty"]
vector_obj = bpy.data.objects["SpeedVector"]
start_arrow = bpy.data.objects["StartVector"]
end_arrow = bpy.data.objects["EndVector"]

for obj in [drone, vector_obj, start_arrow, end_arrow]:
    obj.animation_data_clear()

# Вхідні параметри
V0 = np.array([3.0, 4.0, 5.0])
Vk = np.array([20.0, 40.0, 30.0])
m = 20
dV = 0.1
da = db = dc = 0.01

V0_mag = np.linalg.norm(V0)
Vk_mag = np.linalg.norm(Vk)

# Початкові косинуси
cosa0 = V0[0] / V0_mag
cosb0 = V0[1] / V0_mag
cosc0 = V0[2] / V0_mag

cosak = Vk[0] / Vk_mag
cosbk = Vk[1] / Vk_mag
cosck = Vk[2] / Vk_mag

a0 = np.arccos(cosa0)
b0 = np.arccos(cosb0)
c0 = np.arccos(cosc0)

ak = np.arccos(cosak)
bk = np.arccos(cosbk)
ck = np.arccos(cosck)

nV = int(np.ceil(abs(Vk_mag - V0_mag) / dV))
na = int(np.ceil(abs(ak - a0) / da))
nb = int(np.ceil(abs(bk - b0) / db))
nc = int(np.ceil(abs(ck - c0) / dc))

n = max(nV, na, nb, nc)
k = max(1, n // m)

# Початкові значення
V = V0_mag
a, b, c = a0, b0, c0
frame = 1

# Основний цикл
for i in range(1, n + 1):
    if i % k == 0:
        nV = int(np.ceil(abs(Vk_mag - V) / dV))
        na = int(np.ceil(abs(ak - a) / da))
        nb = int(np.ceil(abs(bk - b) / db))
        nc = int(np.ceil(abs(ck - c) / dc))
        n = max(nV, na, nb, nc)
        k = max(1, n // m)

    denom = max(1, (n - i + 1))
    stepV = (Vk_mag - V) / denom
    stepa = (ak - a) / denom
    stepb = (bk - b) / denom
    stepc = (ck - c) / denom

    V += stepV
    a += stepa
    b += stepb
    c += stepc
```

```
    cosa = np.cos(a)
    cosb = np.cos(b)
    cosc = np.cos(c)

    norm = np.linalg.norm([cosa, cosb, cosc])
    direction = np.array([cosa, cosb, cosc]) / norm
    current = direction * V

    # Анімація
    drone.location = current
    drone.keyframe_insert(data_path="location", frame=frame)

    vector_obj.location = current
    vector_obj.scale = (0.1, 0.1, V / 5.0)
    quat = Vector((0, 0, 1)).rotation_difference(Vector(direction))
    vector_obj.rotation_mode = 'QUATERNION'
    vector_obj.rotation_quaternion = quat
    vector_obj.keyframe_insert(data_path="location", frame=frame)
    vector_obj.keyframe_insert(data_path="scale", frame=frame)
    vector_obj.keyframe_insert(data_path="rotation_quaternion", frame=frame)

    frame += 1

# Перевірка на необхідність вставки фінального Vk
error = np.linalg.norm(current - Vk)
print(f"Final error: {error:.6f}")

if error > 1e-4:
    drone.location = Vk
    drone.keyframe_insert(data_path="location", frame=frame)

    vector_obj.location = Vk
    vector_obj.scale = (0.1, 0.1, Vk_mag / 5.0)
    quat = Vector((0, 0, 1)).rotation_difference(Vector(Vk / Vk_mag))
    vector_obj.rotation_quaternion = quat
    vector_obj.keyframe_insert(data_path="location", frame=frame)
    vector_obj.keyframe_insert(data_path="scale", frame=frame)
    vector_obj.keyframe_insert(data_path="rotation_quaternion", frame=frame)


# Вектори-стрілки початковий/кінцевий
start_arrow.location = Vector(V0)
start_arrow.scale = (0.1, 0.1, V0_mag / 5.0)
start_arrow.rotation_mode = 'QUATERNION'
start_arrow.rotation_quaternion = Vector((0, 0, 1)).rotation_difference(Vector(V0 / V0_mag))

end_arrow.location = Vector(Vk)
end_arrow.scale = (0.1, 0.1, Vk_mag / 5.0)
end_arrow.rotation_mode = 'QUATERNION'
end_arrow.rotation_quaternion = Vector((0, 0, 1)).rotation_difference(Vector(Vk / Vk_mag))
```

ДОДАТОК Б. Код скрипту математичної моделі руху

```
import bpy
import math

# Початкові умови
v0 = 25.0          # початкова швидкість (м/с)
alpha_deg = 60.0   # кут в градусах
z0 = 5.0           # початкова висота (м)
g = 9.81           # прискорення вільного падіння

alpha = math.radians(alpha_deg)
cos_a = math.cos(alpha)
sin_a = math.sin(alpha)

# Об'єкт для руху
empty = bpy.data.objects["Empty"]
empty.animation_data_clear()

# Час польоту (аналітично)
under_sqrt = (v0 * sin_a)**2 + 2 * g * z0
t_flight = (v0 * sin_a + math.sqrt(under_sqrt)) / g


fps = 24
frames = int(t_flight * fps)


for frame in range(frames + 1):
    t = frame / fps
    x = v0 * cos_a * t
    z = z0 + v0 * sin_a * t - 0.5 * g * t**2


    if z < 0:
        z = 0
        empty.location = (x, 0, z)
        empty.keyframe_insert(data_path="location", frame=frame)
        break

    empty.location = (x, 0, z)
    empty.keyframe_insert(data_path="location", frame=frame)
```