Харківський національний університет імені В.Н.Каразіна
Факультет математики і інформатики
Кафедра прикладної математики

Кваліфікаційна робота
бакалавра

на тему “*Математичне моделювання процесу швидкого навантаження альпіністської мотузки*”

Виконав:
студент групи МП-41
4 курсу
спеціальність 113 – прикладна математика
освітньо-професійна програма
“Прикладна математика”
Собур М. С.
Науковий керівник: кандидат ф-м. наук, доцент
Послався́кий С.О.
Рецензент: кандидат техн. наук, доцент
Духопельников С.В.

Харків - 2023 рік

Анотації

Собур М.С. “*Математичне моделювання процесу швидкого навантаження альпіністської мотузки*”

У роботі розглянута двовимірна модель мотузкової системи, що складається з базової горизонтальної мотузки і прикріпленої до неї посередині мотузки з вантажем на іншому кінці. Наведено огляд відомих реологічних моделей, які використовуються у дослідженнях подібних систем. Виведені рівняння для одновимірного і двовимірного випадків. Розроблена програма на мові програмування Python, що моделює рух системи у площині.

Sobur. M. S. “*Mathematical modeling of the rapid loading process of a climbing rope*”

Bachelor’s degree thesis presents a two-dimensional model of a system consisting of a basic horizontal rope and a rope attached to it in the middle with a load at the other end. An overview of existing rheological models used in studies of such systems is given. Equations for both one-dimensional and two-dimensional cases are derived. A Python program is developed to simulate the motion of the system in the plane.

Зміст

	Анотації	**2**
	Вступ	**4**
1.	**Одновимірна модель мотузкової системи**	**9**
2.	**Двовимірна модель мотузкової системи**	**11**
3.	**Комп'ютерне моделювання**	**19**
4.	**Лістинги програми**	**25**
	Висновки	**34**
	Список використаних джерел	**35**

Вступ

Сучасні мотузкові системи мають різноманітні характеристики. Насамперед, вони мають пружні властивості, що описуються поведінкою звичайної пружини (Рис. 1) і, відповідно, законом Гука при малих деформаціях, а також в'язкі властивості, що описуються поведінкою демпфера (Рис. 2) і законом в'язкого тертя Ньютона.

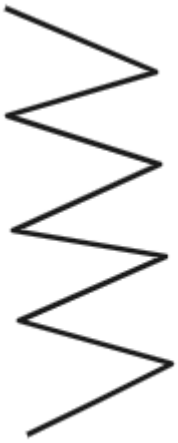


Рис.1. Схематичне зображення пружини.

Рис. 2. Схема демпфера.

Як відомо, закон Гука при малих деформаціях виглядає наступним чином:

$$F = - kx,$$

де k – жорсткість пружини, x – деформація.

Закон в'язкого тертя Ньютона має наступний вигляд:

$$\sigma = \eta \frac{d\varepsilon}{dt},$$

де σ – напруга (можна вважати цей показник силою), що діє на в'язкий елемент, η – коефіцієнт в'язкості речовини, ε – деформація.

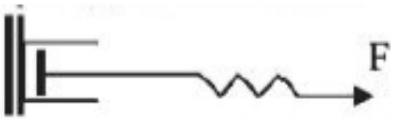
Тобто фактично остання рівність означає, що напруга лінійно залежить від швидкості зміни деформації.

Демпфер являє собою циліндр (або трубку) заповнену високов'язкою речовиною, у якій при навантаженнях рухається поршень. В залежності від

конкретної задачі це можуть бути різні речовини. Зазвичай використовують синтетичні масла (наприклад, силіконове), воду з гліколем, гідравлічні рідини. Серед останніх є такі види рідин як: магнітореологічна та електрореологічна. В'язкість першого виду може значно змінюватися під дією магнітного поля, а в'язкість другого – під дією електричного. Детальніше можна ознайомитись з цим у відповідних джерелах в Інтернеті.

Комбінуючи різні види з'єднань цих двох основних елементів, можна за допомогою диференціальних рівнянь описувати найрізноманітніші за складністю в'язко-пружні системи. Далі розглянемо деякі основні відомі моделі, які були побудовані як комбінація послідовних і паралельних з'єднань цих двох елементів.

1. Реологічна модель Максвелла



Являє собою послідовне з'єднання в'язкого і пружного елементів. При такому типі з'єднання напруга (або сила) у кожному з елементів є однаковою, а загальна деформація системи дорівнює сумі деформацій. Рівняння загальної деформації системи виглядає наступним чином:

$$\frac{d\varepsilon}{dt} = \frac{1}{E}\frac{d\sigma}{dt} + \frac{\sigma}{\eta},$$

де E – модуль пружності Юнга (характеристика пружини, подібна до пружності), σ – загальна напруга у системі, η – в'язкість речовини демпфера.

Для різних задач, пов'язаних з моделюванням поведінки в'язко-пружних систем може бути корисною узагальнена модель Максвелла, що являє собою паралельне з'єднання зазначеної вище моделі. Вона зображена на рисунку нижче.

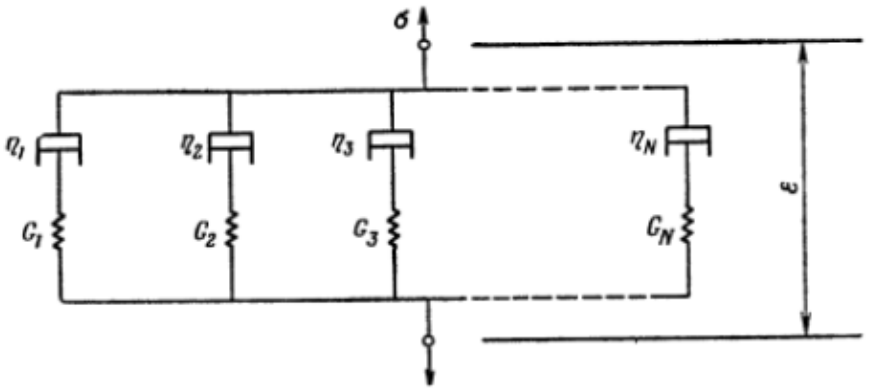
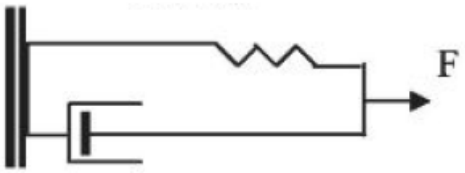


Рис. 3. Узагальнена модель Максвелла [3].

2. Модель Кельвіна-Фойгта



Являє собою паралельне з’єднання пружини і демпфера. Напруга у цій системі дорівнює сумі напруг пружної деформації і деформації у в’язкому елементі. Рівняння зв’язку між напругою і деформацією виглядає наступним чином:

$$\sigma = E\varepsilon + \eta\frac{d\varepsilon}{dt}$$

Ця модель, звичайно, також допускає узагальнення зображене на рисунку нижче. Наведена модель у поєднанні з пружиною або набором інших моделей може чудово описувати властивості кісткової тканини живих організмів.

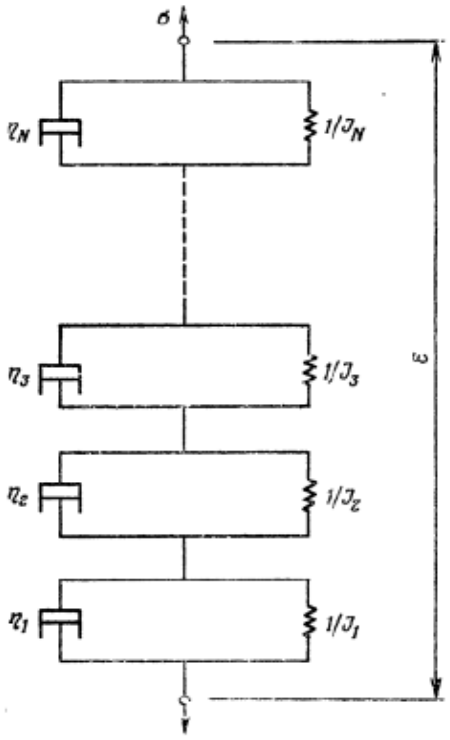


Рис. 4. Узагальнена модель Кельвіна-Фойгта [3].

3. Модель навантаженого стандартного лінійного тіла (SLS, Standart linear solid model)

Ця модель являє собою паралельне з’єднання пружини і моделі Максвелла.

Система диференціальних рівнянь, що описує рух вантажа у цій моделі виглядає наступним чином:

$$m\frac{d^2y}{dt^2} + k_1y + k_2(y - y_i) = mg$$

$$c\frac{dy_i}{dt} + k_2(y_i - y) = 0$$

Ці рівняння можна описувати у термінах класичної механіки, тобто спираючись на поняття потенціальних і не потенціальних складових сил, що діють на систему. Або у термінах теорії в’язко-пружності, тобто використовуючи рівняння, що містять зв’язки між напругами, деформаціями, коефіцієнтами в’язкості і пружності.

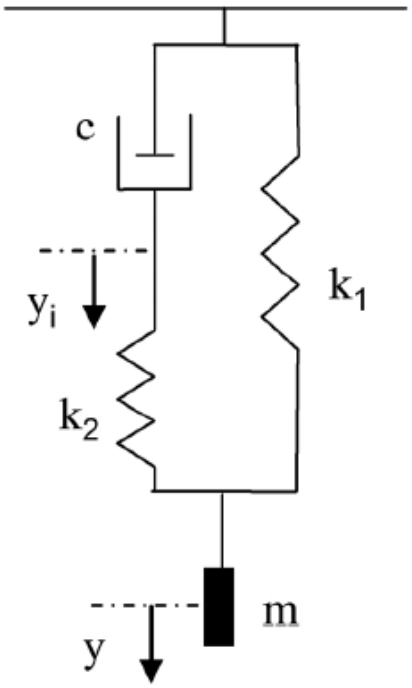


Рис. 5. SLS-модель навантаженої мотузки [1].

Для розуміння того, з яких міркувань були отримані вказані рівняння можна ознайомитися з відповідною літературою (див. [1]-[3]).

Вказані вище моделі мають велике прикладне значення. Окрім безпосередньо мотузкових систем, в’язко-пружні моделі використовуються у різних галузях науки та інженерії, де необхідно враховувати деформаційні властивості матеріалів та середовищ. Наведемо деякі з таких областей та проблем, де використовуються ці моделі.

1. Механіка та матеріалознавство: моделі застосовуються для опису деформацій та поведінки різних матеріалів, таких як полімери, композити, метали та ін. Ці моделі допомагають передбачити довготривалі деформації, реологічні властивості та поведінку матеріалів під різними навантаженнями.

2. Геотехніка: у геотехніці вони використовуються для дослідження та моделювання деформацій ґрунтів та порід. Моделі дозволяють передбачити поведінку фундаментів, стабільність схилів, деформації при гідро- та термонавантаженнях, а також тривалі процеси, пов'язані з напругами та деформаціями в ґрунтах.

3. Біомеханіка та медицина: в'язко-пружні моделі знаходять застосування в біомеханіці та медичних дослідженнях, де необхідно аналізувати деформації тканин та органів людини. Ці моделі допомагають зрозуміти взаємодію механічних навантажень з біологічними тканинами та оптимізувати процеси відновлення та реабілітації.

4. Реологія та текстильна промисловість: У текстильній промисловості реологічні моделі використовуються для дослідження та передбачення поведінки текстильних матеріалів, таких як волокна, нитки та тканини. Це дозволяє покращити процеси виробництва і якість текстильних виробів, а також розробляти нові матеріали.

5. Гідродинаміка та нафтогазова промисловість: моделювання руху рідин та газів у різних інженерних системах. Це дозволяє передбачати перебіг нафти, газу та води у свердловинах, трубопроводах і резервуарах, а також покращувати процеси видобутку, транспортування та зберігання цих ресурсів.

1. Одновимірна модель мотузкової системи

Розглянемо модель, що описує поздовжні хвилі у мотузці. Спочатку нагадаємо, що існують два основні види хвиль: поперечна і поздовжня. Якщо при розповсюдженні хвилі по середовищу (у даному випадку по мотузці) частинки самого середовища зміщуються у тому самому напрямку, що і хвиля, то така хвиля називається поздовжньою. Якщо ж частинки середовища рухаються у напрямку перпендикулярному до напрямку руху самої хвилі, то така хвиля називається поперечною. У природі частіше за все хвилі, що виникають у різних середовищах являють собою комбінацію цих елементарних типів хвиль.

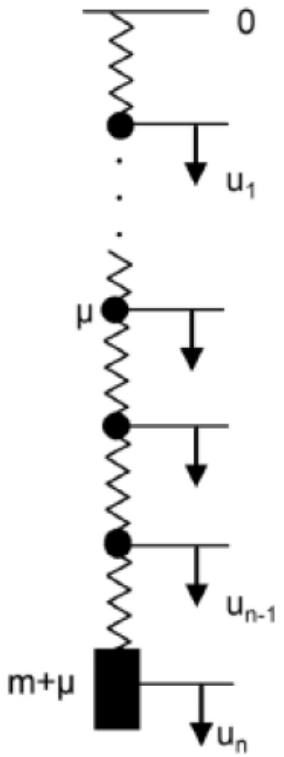


Рис. 6. Модель, що описує поздовжні хвилі у мотузці [1].

На рис. 6 зображена модель мотузки-поводка. Вона являє собою послідовне з'єднання точкових мас μ пружинами. Верхній кінець цієї моделі закріплений і не здійснює ніяких рухів (хоча у моделі, яку ми розглянемо далі це буде точка кріплення, що здійснюватиме вертикальні рухи). На нижньому кінці закріплений вантаж (вважаємо, що він має масу m).

Рівняння руху системи можна вивести за допомогою рівнянь Лагранжа або використовуючи Другий закон Ньютона і Закон Гука.

В результаті отримаємо наступну систему рівнянь:

$$\mu\frac{d^2u_i}{dt^2} + k(u_i - u_{i+1}) + k(u_i - u_{i-1}) = \mu g \qquad (1.1)$$

$$m\frac{d^2u_n}{dt^2} + k(u_n - u_{n-1}) = mg \qquad (1.2)$$

Рівняння (1.1) описує рух i-ї точкової маси ($i = 1, \cdots, n-1$), рівняння (1.2) – рух самого вантажа.

Роблячи заміну зміних можна позбутися статичної складової у правій частині рівняння, тобто можна отримати систему однорідних диференціальних рівнянь.

Також одним з можливих підходів для опису поведінки цієї механічної системи є розглядання так званого хвильового рівняння (1.3), яке являє собою рівняння у частинних похідних. Для цього необхідно розглянути граничний перехід при $n \to \infty,\ m \to 0$. Таким чином, довжина кожної з пружинок прямуватиме до 0. При такому підході ця мотузкова система розглядається як неперервне одновимірне середовище, у якому розповсюджуються хвилі (або як функція координати і часу). Аналітичний розв'язок такого типу рівнянь зазвичай шукають за допомогою методу розділення змінних, попередньо задаючи початкові та крайові умови. Хвильове рівняння має наступний вигляд:

$$\frac{\partial^2 u}{\partial t^2} - a^2 \frac{\partial^2 u}{\partial x^2} = f(x,t), \quad (1.3)$$

де a – фазова швидкість, $u(x,t)$ – невідома функція координати і часу, $f(x,t)$ – відома наперед функція. Опис виводу цього рівняння можна подивитися у відповідній літературі з рівнянь у частинних похідних [6].

Як вже було зазначено вище, система рівнянь (1.1), (1.2) потрібна нам для опису поведінки мотузки-поводка у нашій основній моделі.

2. Двовимірна модель мотузкової системи

Розглянемо нашу основну модель. Маємо дві фіксовані на одному рівні по висоті точки A і B. Ці дві точки з'єднані мотузкою, яку ми також моделюємо як ланцюжок послідовно з'єднаних пружинами рівномірно розподілених точкових мас m. Цю мотузку ми називаємо базовою і вона має попередній натяг (тобто довжина кожної з пружинок, що з'єднує пару мас строго більша натуральної довжини цих пружинок). Посередині у точці C до цієї мотузки під'єднана інша мотузка, яку ми називаємо мотузкою-поводком. Ця мотузка являє собою модель, розглянуту у попередньому пункті з пружинами жорсткості k_2, яка менша ніж

жорсткість пружинок базової мотузки. Вважаємо, що всі точки мотузки-поводка здійснюють лише вертикальні рухи.

Стверджуємо, що всі точки системи здійснюють рухи лише у площині (тобто кожна точка має дві координати). Вводимо систему координат наступним чином: горизонтальна вісь Ox направлена вздовж базової мотузки у напрямку зростання значень координати від точки A до точки B. За початок координат вважаємо точку C. Вертикальна вісь Oy направлена вздовж мотузки-поводка, а напрямок зростання значень цієї координати від точки C до точки D (остання точка являє собою вантаж). Для спрощення вважаємо, що точки A і B розташовані симетрично відносно точки C, і відповідно, самі точкові маси бази також.

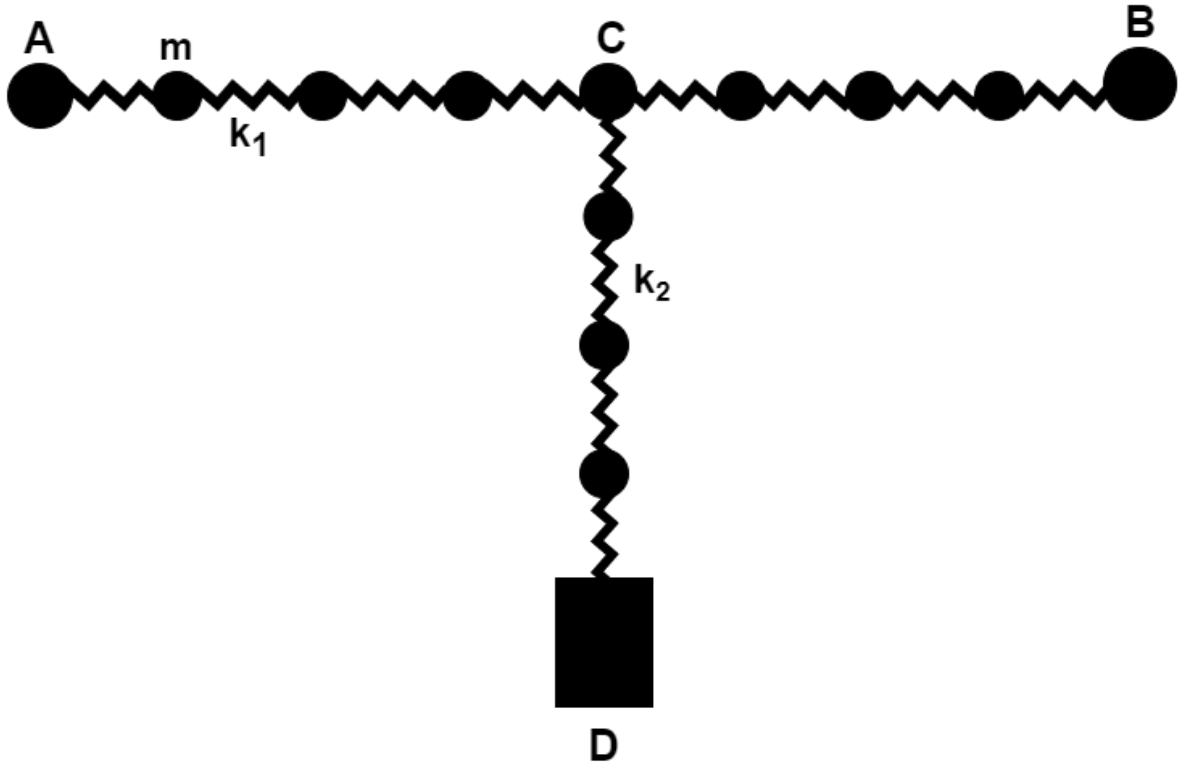


Рис. 7. Двовимірна модель з базовою мотузкою.

Введемо деякі позначення. Нехай у нашій системі є по n рухомих точок справа і зліва від точки C і s точок між точкою C і D. Тоді загальна кількість точок $2n + s + 2$. Кожна точка має дві координати x, y, а тому маємо $2 * (2n + s + 2)$ степенів свободи у системі. Кожну точку базової мотузки позначимо як m_i з відповідними координатами (x_i, y_i).

Нехай $l_0^{\ 1}$ – натуральна (ніяк не деформована) довжина кожної з пружин базової мотузки. Для спрощення вважаємо, що всі пружини однієї і тієї ж самої мотузки є однаковими за характеристиками (перш за все пружність та натуральна довжина).

$l_0^{\ 2}$ – натуральна довжина будь-якої з пружин вертикальної мотузки.

$l_{i,i+1}(t) = \sqrt{(y_i - y_{i+1})^2 + (x_i - x_{i+1})^2}$ – довжина пружини базової мотузки у деякий момент часу t між точками m_i та m_{i+1}.

$\Delta l_{i,i+1}(t) = l_{i,i+1}(t) - l^1_{\ 0}$ – подовження пружини бази між масами m_i та m_{i+1} у момент часу t.

k_1, k_2– жорсткості пружин бази і мотузки-поводка відповідно.

Як було зазначено вище, вважаємо що $l_{i,i+1}(0) = x_{i+1}(0) - x_i(0) > l_0^{\ 1}$. Тобто у початковий момент часу є попередній натяг бази.

$\alpha_i(t)$ – кут між пружиною бази з правим кінцем у точці m_{i+1} у деякий момент часу t і напрямком горизонтальної осі.

$\tilde{y}_i$ – позначення для координат y точок вертикальної мотузки.

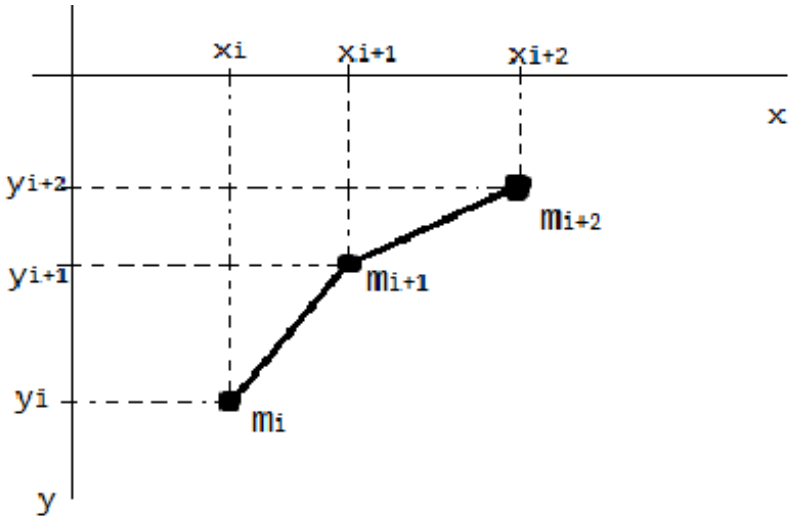


Рис. 8. Схематичне зображення фрагменту мотузкової системи.

Проектуючи силу пружності Гука на відповідні осі, отримаємо вирази для кожної окремої точки m_i.

$$F_{y_i} = -k_1 \Delta l_{i,i+1} \sin\alpha_i = -k_1 (l_{i,i+1} - l^1_0) \frac{y_i - y_{i+1}}{l_{i,i+1}}$$

$$F_{x_i} = -k_1 \Delta l_{i,i+1} \cos\alpha_i = -k_1 (l_{i,i+1} - l^1_0) \frac{x_i - x_{i+1}}{l_{i,i+1}}$$

Або після розкриття дужок отримаємо:

$$F_{y_i} = -k_1 (y_i - y_{i+1})\left(1 - \frac{l^1_0}{l_{i,i+1}}\right)$$

$$F_{x_i} = -k_1 (x_i - x_{i+1})\left(1 - \frac{l^1_0}{l_{i,i+1}}\right)$$

Користуючись Другим законом Ньютона і законом Гука, а також тим, що пружини з'єднані послідовно, отримуємо систему рівнянь:

$$m_i \frac{d^2 y_i}{dt^2} = -k_1 (y_i - y_{i+1})\left(1 - \frac{l^1_0}{l_{i,i+1}}\right) - k_1 (y_i - y_{i-1})\left(1 - \frac{l^1_0}{l_{i-1,i}}\right) \qquad (2.1)$$

$$m_i \frac{d^2 x_i}{dt^2} = -k_1 (x_i - x_{i+1})\left(1 - \frac{l^1_0}{l_{i,i+1}}\right) - k_1 (x_i - x_{i-1})\left(1 - \frac{l^1_0}{l_{i-1,i}}\right) \qquad (2.2)$$

$$i = 1, 2, \cdots, n,\ n+2, \cdots, 2n+1$$

Вона описує рух точок саме базової мотузки. Окремо детальніше розглянемо рівняння руху точки C. На цю точку діють два види сил. З одного боку сили натягу напрямлені від неї вздовж базової мотузки до кінців, а з іншого боку сила направлена вниз вздовж мотузки-поводка до точки кріплення вантажа D. Тому у її рівнянні руху будуть присутні як доданки у цій системі так і доданок такого вигляду як у формулі (1.1). Тому маємо наступне рівняння (2.3):

$$m_C \frac{d^2 y_C}{dt^2} = -k_1 (y_C - y_{n+2})\left(1 - \frac{l^1_0}{l_{C,n+2}}\right) - k_1 (y_C - y_{n-1})\left(1 - \frac{l^1_0}{l_{n-1,C}}\right) - k_2 (y_C - \tilde{y}_1)$$

де m_C, y_C – маса і координата за y точки C відповідно. Додамо також, що зазвичай, ця точка (у реальних системах – ролик) має деяку не малу масу, якою не можна нехтувати (вона може сягати порядку 0.5-1 кг).

Тепер розглянемо рівняння координат y для точок вертикальної мотузки. Варто зазначити, що вони мають деякі специфічні моменти у порівнянні з формулами отриманими для опису руху моделі з пункту 1.

Особливість полягає у тому, що нас цікавить тільки стан мотузки, при якому відбувається саме розтягнення (тобто додатня деформація). Це пояснюється тим, що тільки за цієї умови у мотузці виникає напруга, тобто сили натягу, що змушують її повернутися у стан спокою і мотузка у цьому сенсі може добре моделюватися пружиною. І в той же час ми не повинні розглядати від'ємну деформацію, тобто коли мотузка стискається. У цьому випадку не виникає жодних сил спротиву, тому що мотузка змінює свою прямолінійну форму (тобто синусоїдно вигинається) і пружина не підходить для опису мотузки у такому стані, оскільки при стисканні пружини вона хоче відновити свою довжину, яку вона мала у стані спокою. Тому у аналітичному вигляді це можна виразити наступним чином: $F_{\text{пруж.}} = -k_2\Delta l, \Delta l > 0,\ F_{\text{пруж.}} = 0, \Delta l < 0$. Тобто фактично це означає, що сила пружності, яка діє у мотузці-поводку має розривний характер.

Приймаючи до уваги цей факт і використовуючи вже відомі рівняння для одновимірної моделі, отримуємо:

$$m_j \frac{d^2\tilde{y}_j}{dt^2} = -k_2(\tilde{y}_j - \tilde{y}_{j+1}) - k_2(\tilde{y}_j - \tilde{y}_{j-1}) \qquad (2.4)$$

$$j = 1, 2, \cdots, s$$

По-друге, ми звичайно вважаємо, що у точці D маса (у формулі нижче ми виділили її більшою буквою M) значно більша за маси точок мотузок.

Рівняння для точки D:

$$M\frac{d^2\tilde{y}_D}{dt^2} = Mg - k_2(\tilde{y}_D - \tilde{y}_s) \qquad (2.5)$$

Тобто отримали систему нелінійних диференціальних рівнянь другого порядку з постійними коефіцієнтами.

Задаючи початкові умови і використовуючи чисельні методи можна отримати наближений розв'язок цієї системи. Варто зазначити, що на рисунку 7. зображена модель у момент часу $t = 0$, коли вантаж вже рухається вниз вільно падаючи з висоти. Задаючи початкові умови, вважаємо, що мотузка-поводок у цей момент часу має довжину $(s + 1)l_0^2$. Тобто мотузка-поводок у даний момент часу дорівнює сумі усіх її не натягнутих пружин. Це важливо, оскільки нас цікавить поведінка системи, коли в ній виникають сили натягу. Попереднього натягу у мотузки-поводка немає. Точка, з якої відбувається падіння розташовується вище за базову мотузку. Для спрощення моделювання розглядатимемо випадок, коли вона знаходиться вертикально над точкою C. Також при моделюванні вважається, що на точки бази не діє сила тяжіння до моменту натягу мотузки-поводка. Зазначимо, що ми також нехтуємо тертям базової мотузки з повітрям при поперечному русі. Воно може бути вагомим фактором, але модель від цього буде значно ускладнюватися.

З вигляду системи зрозуміло, що вона має симетричний вигляд відносно серединної точки C. З цього випливає, що для опису системи достатньо обмежитись розглядом половини рівнянь, яка описує рух точок по один бік від цієї точки. У загальному випадку можна розглядати і несиметричний вигляд. Тоді рівняння у симетричних точок будуть різними внаслідок того, що по різному будуть змінюватися кути α_i при русі системи.

Отже, початкові умови для задачі Коші матимуть наступний вигляд:

$v_D = \sqrt{2gh}$ – швидкість руху вантажа (точка D). g – прискорення вільного падіння на Землі, h – висота, з якої падає тіло (можно обирати будь-яку).

v_C, v_i – початкові швидкості точки C і точок базової мотузки.

$t_0 = 0$ – момент часу, коли мотузка-поводок CD випрямлена, але її довжина дорівнює натуральній довжині (тобто мотузка ще не розтягнута).

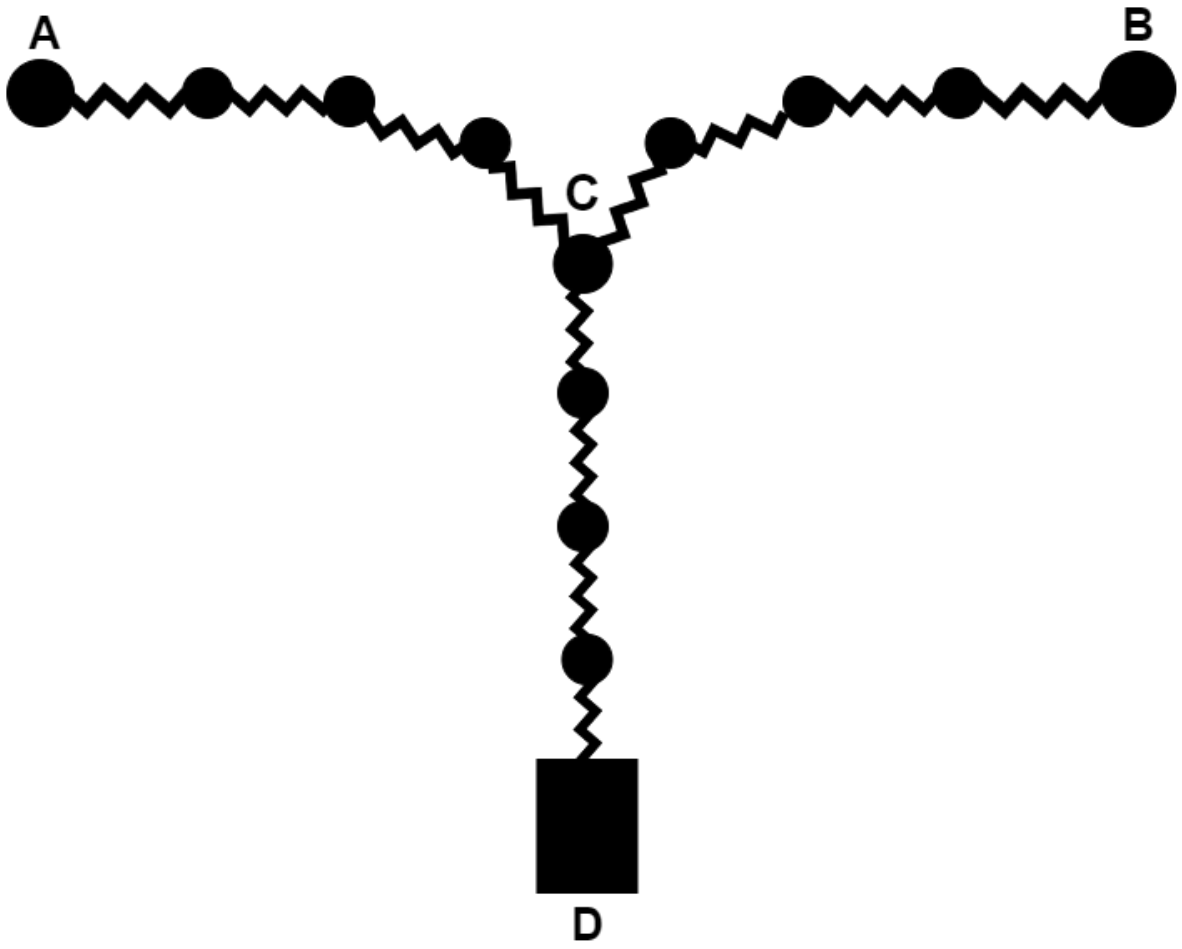


Рис. 9. Двовимірна модель у натягнутому вигляді.

Узагальнення рівнянь (2.1) - (2.5)

Як відомо, при паралельному з’єднанні елементів напруга, що виникає в системі дорівнює сумі напруг, на кожному з елементів окремо, а деформація кожного з елементів дорівнює загальній деформації системи.

Приймаючи до уваги цей факт і додаючи до кожної з пружин у системі паралельно з’єднаний демпфер як показано на зображенні фрагменту нижче, отримуємо наступне узагальнення для рівнянь (2.1) - (2.5):

Рис. 10. Фрагмент узагальненої моделі.

$$m_i\frac{d^2y_i}{dt^2} + \eta_1(1 - \frac{l^1{}_0}{l_{i,i+1}})\frac{dy_i}{dt} = -k_1(y_i - y_{i+1})(1 - \frac{l^1{}_0}{l_{i,i+1}}) - k_1(y_i - y_{i-1})(1 - \frac{l^1{}_0}{l_{i-1,i}})$$

$$m_i\frac{d^2x_i}{dt^2} + \eta_1(1 - \frac{l^1{}_0}{l_{i,i+1}})\frac{dx_i}{dt} = -k_1(x_i - x_{i+1})(1 - \frac{l^1{}_0}{l_{i,i+1}}) - k_1(x_i - x_{i-1})(1 - \frac{l^1{}_0}{l_{i-1,i}})$$

$$i = 1, 2, \cdots, n,\ n + 2, \cdots, 2n + 1$$

Позначимо для зручності $L_{i,i+1} = (1 - \frac{l^1{}_0}{l_{i,i+1}})$, тоді коефіцієнт при похідній:

$$K_C = (\eta_1(L_{C,n+2} + L_{n-1,C}) + \eta_2),$$

$$m_C\frac{d^2y_C}{dt^2} + K_C\frac{dy_C}{dt} = -k_1(y_C - y_{n+2})L_{C,n+2} - k_1(y_C - y_{n-1})L_{n-1,C} - k_2(y_C - \tilde{y}_1)$$

$$m_j\frac{d^2\tilde{y}_j}{dt^2} + \eta_2\frac{d\tilde{y}_j}{dt} = -k_2(\tilde{y}_j - \tilde{y}_{j+1}) - k_2(\tilde{y}_j - \tilde{y}_{j-1})$$

$$j = 1, 2, \cdots, s$$

$$M\frac{d^2\tilde{y}_D}{dt^2} + \eta_2\frac{d\tilde{y}_D}{dt} = Mg - k_2(\tilde{y}_D - \tilde{y}_s),$$

де η_1, η_2 – коефіцієнти в’язкості демпферів бази і поводка відповідно.

Отже, ми отримали узагальнені рівняння (2.6) - (2.10).

У кінці цього розділу додамо, що для створення натягу базової мотузки у реальних системах роуп-джампінгу у точках кріплення може бути розміщена котушка, через яку перекидають цю мотузку і потім її кінець намотують на ще одну котушку розміщену нижче відносно цих точок.

3. Комп’ютерне моделювання

У цьому розділі ми надамо результати комп’ютерного моделювання руху нашої моделі. Програма була розроблена за допомогою мови програмування Python, а також фізичної бібліотеки для симуляції руху тіл Pymunk і бібліотеки для розробки мультимедійних додатків з графічним інтерфейсом Pygame.

Було проведено багато експериментів з кількістю точок у системі, їх розміщенням і зміною інших параметрів. Код побудований таким чином, що дослідник-експериментатор може задаючи відповідні параметри вручну отримувати власні результати, тобто певну поведінку системи і форму мотузки.

Спочатку ми проводили експерименти просто підбираючи параметри емпіричним шляхом. Значення деяких з них ми додали у коментарі в коді. Але для отримання потрібної нам поведінки потрібно фіксувати характеристики мотузок в цілому, а змінювати кількість точкових мас поступово збільшуючи їх. Тим самим збільшиться і кількість пружинок у системі. Особливість задання параметрів при такому підході в тому, що маса однієї точки залежить від наперед заданої маси мотузки (а не є довільним числом). А саме треба розділити масу всієї мотузки на кількість точок бази (без центральної точки кріплення) і таким чином ми отримаємо рівномірний розподіл мас. А при збільшенні пружин і точок ми повинні отримувати більш точне наближення поведінки до тої, яку ми можемо спостерігати у житті. Також зазначимо, що при послідовному з’єднанні пружин жорсткість “маленької” пружини (тобто однієї пружини між парою точок) більша за жорсткість “великої” пружини у кількість разів, що дорівнює кількості точок. “Великою” можна вважати всю мотузку, що моделюється у програмі пружиною від лівої точки кріплення до правої, або половину такої мотузки до рухомого центру. Пояснити такі співвідношення можна з наступних міркувань. Розглянемо пружину фіксованої довжини і подіємо на неї деякою силою F. Отримаємо деяку її додатну деформацію. Далі розглянемо деяку невелику кількість пружин меншої довжини з’єднаних послідовно з додатковою умовою, що вони мають однакову між собою жорсткість, а їхня сумарна довжина у стані спокою дорівнює довжині великої

пружини. Тоді при прикладанні цієї самої сили F отримаємо, що сума деформацій маленьких пружин (які є рівними) дорівнює деформації великої пружини. А тоді із закону Гука отримуємо, що жорсткість малої пружини більша за жорсткість великої рівно у кількість малих пружин. Нижче наведемо знімки у різні моменти часу.

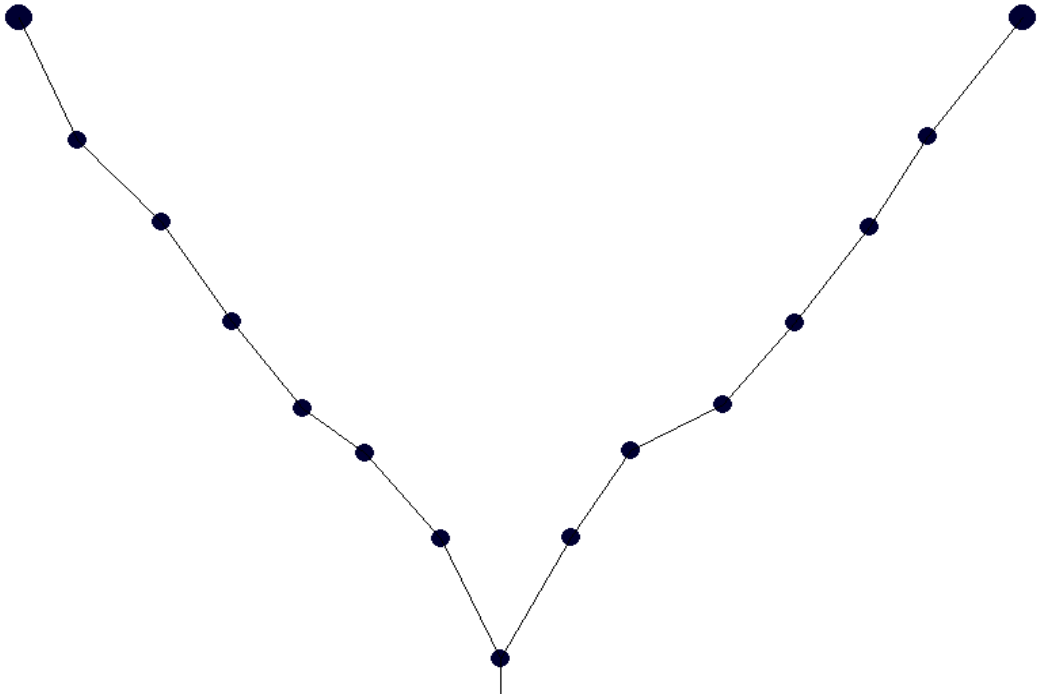


Рис.11. Знімок 1 вікна екрану у момент руху системи.

Як можна побачити на цьому рисунку дійсно спостерігається симетрія відносно вертикальної вісі, що проходить через центральну точку. Додамо ще, що точка з вантажем знаходиться нижче поза знімком вікна Pygame. За бажання у програмі можна розглядати і несиметричний випадок. Під цим ми розуміємо той факт, що задаючи непарну кількість точок бази можна отримати зміщення центральної точки в бік. Серед початкових параметрів, які попередньо задаються присутні параметри зміщення вагової точки. Таким чином можна розглянути випадок, коли тіло починає вільне падіння не рівно над центральною точкою.

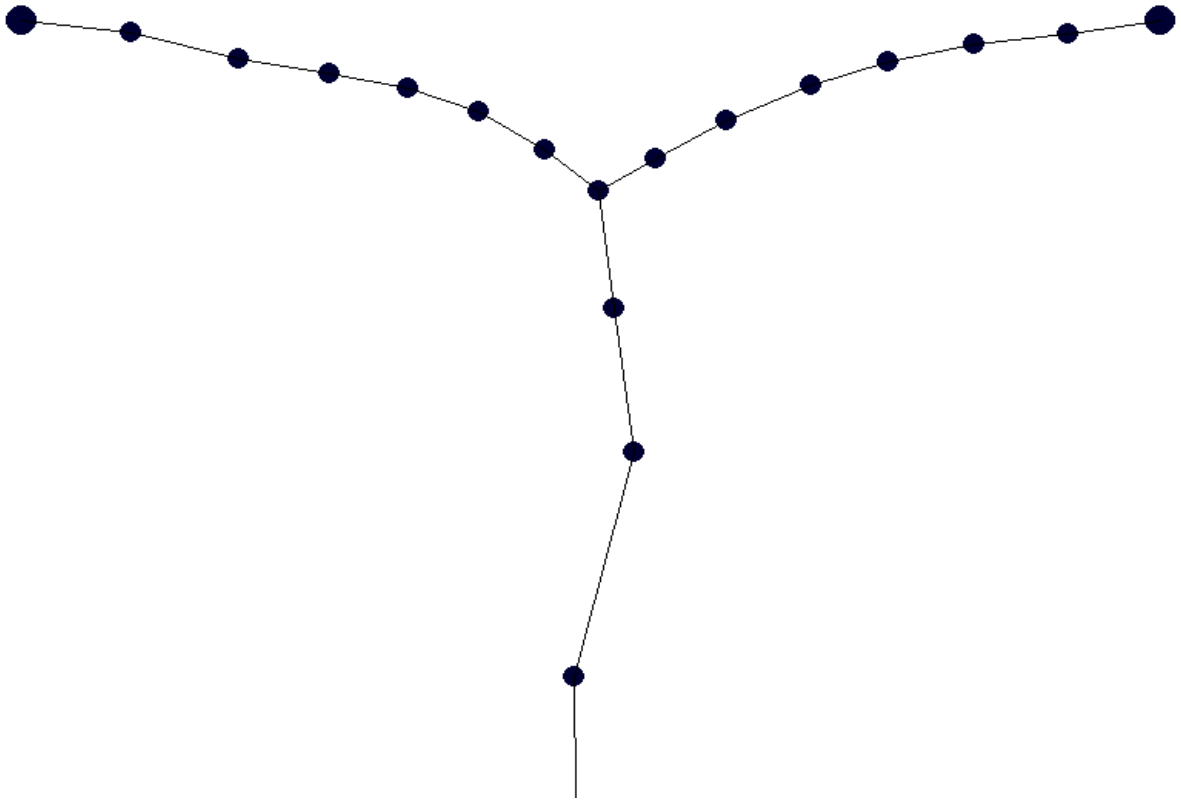


Рис.12. Знімок 2 вікна екрану у момент руху системи.

На рисунку вище можна побачити як мотузка-поводок повертається з розтягнутого стану і тому відбувається зміщення вверх і центральної точки бази.

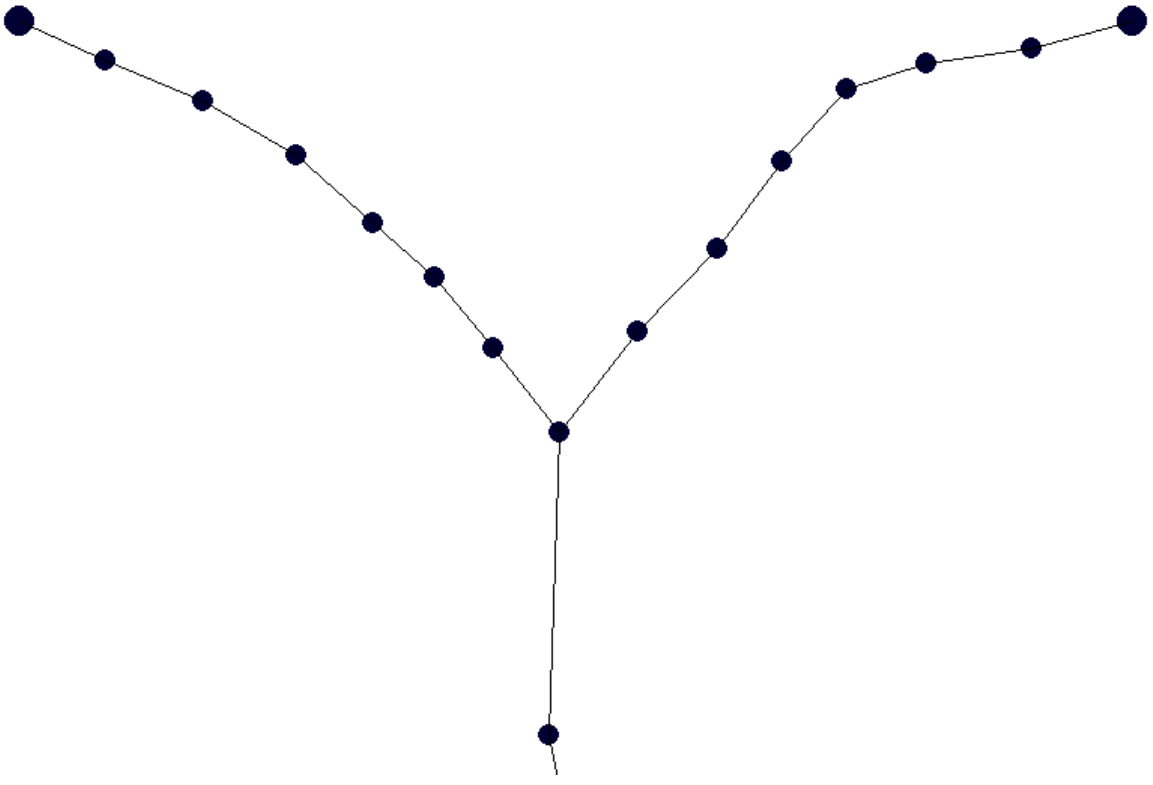


Рис.13. Знімок 3 екрану у більш пізній момент часу.

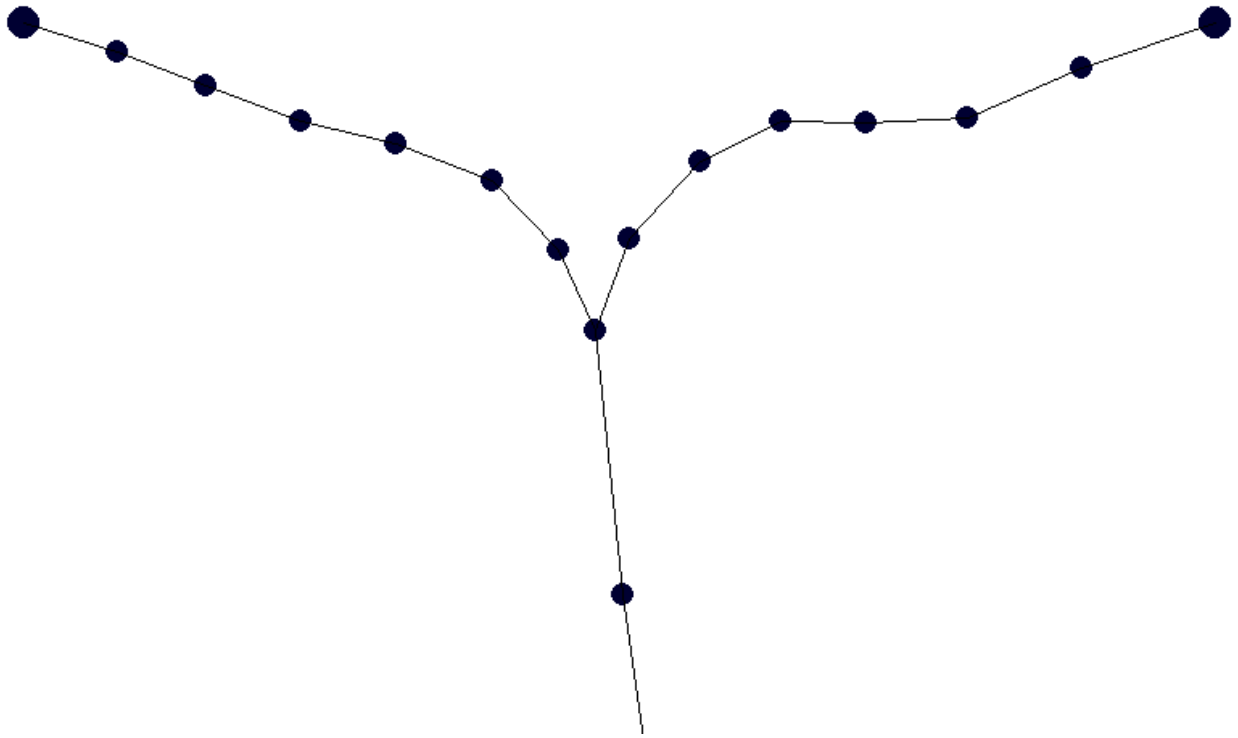


Рис.14. Знімок екрану 4.

На знімках 3 і 4 можна побачити наявність асиметрії у формі мотузки відносно центральної точки. Також на останньому знімку зправа можна явно помітити розповсюдження поперечної хвилі (фрагмент дуги). Вона має достатньо велику амплітуду і оскільки, як вже було згадано вище, вона розповсюджується значно повільніше за хвилі поздовжні, тому цей знімок є наглядною демонстрацією впливу цього типу хвиль на форму мотузки.

Нижче розглянемо серію знімків зроблених підряд з інтервалом в одну секунду. Можна звернути увагу на зміну напрямку опуклості фрагментів моделі зліва і зправа від центру в залежності від ступеня натягнутості мотузки-поводка.

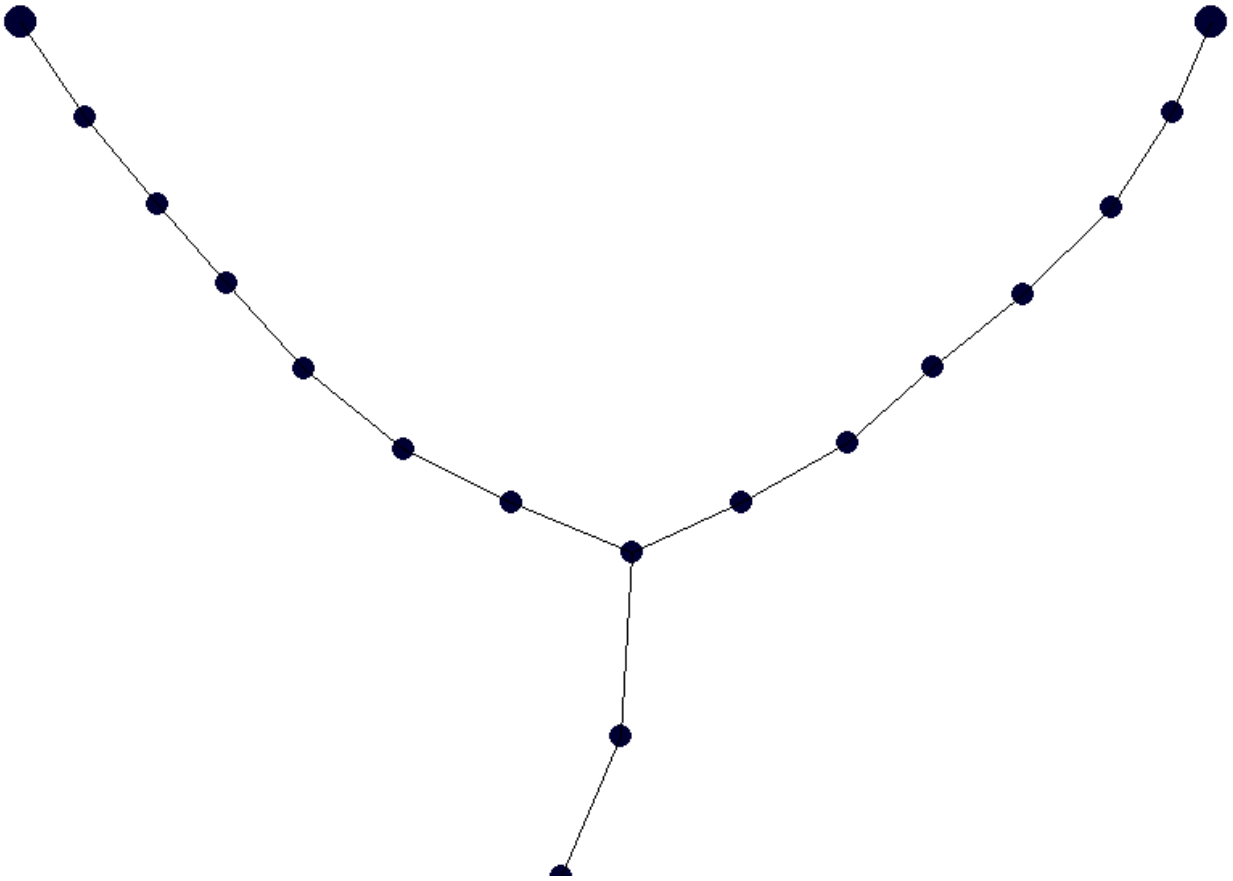


Рис.15. Послідовний знімок екрану 5 у перший момент часу.

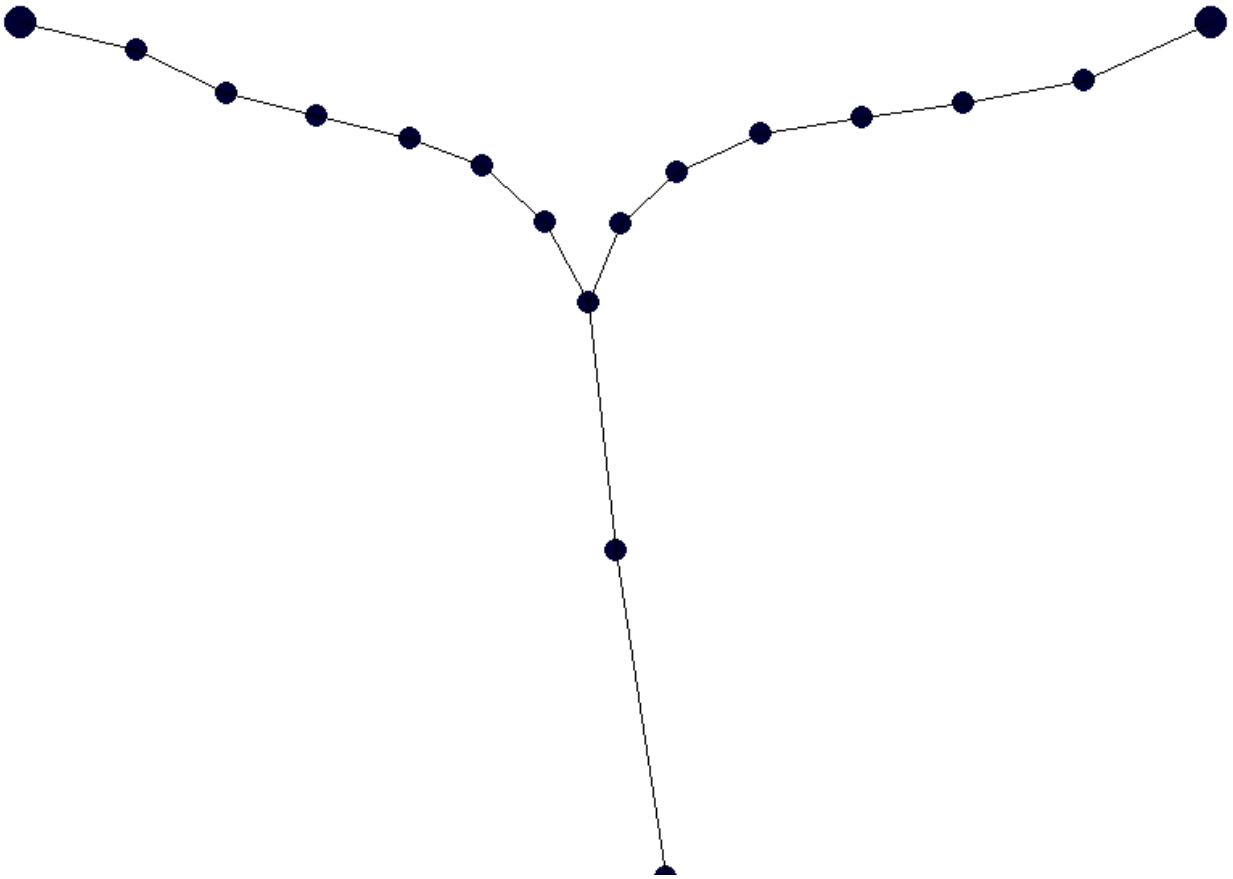


Рис.16. Послідовний знімок екрану 6 у другий момент часу (через секунду).

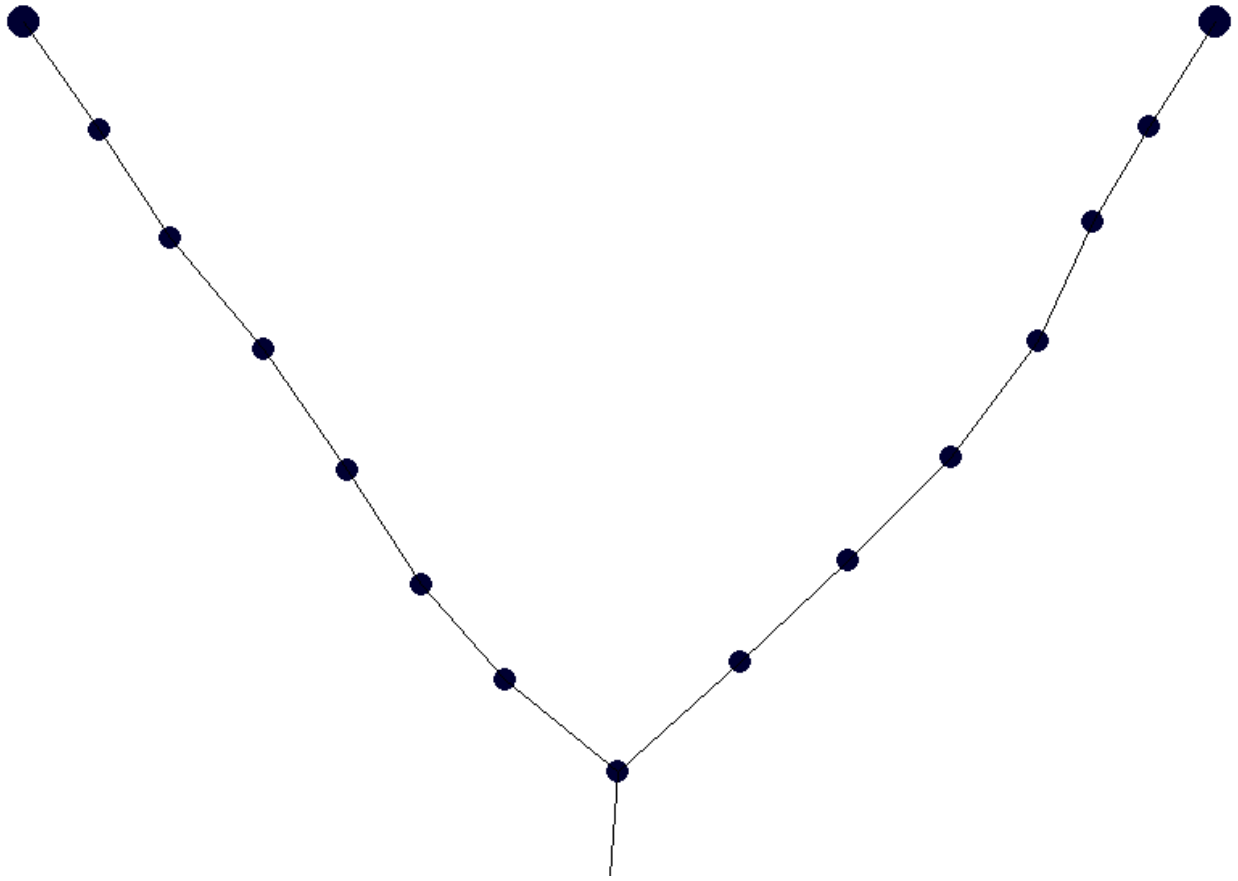


Рис. 17. Третій послідовний знімок 7 через дві секунди від першого.

Ця послідовна серія з трьох знімків дає нам змогу спостерігати динаміку зміни системи, тобто її еволюцію протягом деякого проміжку часу. Серед іншого можна помітити значне подовження пружин поводка у порівнянні з пружинами бази. Цей ефект пояснюється тим, що ми спеціально розглядали показник жорсткості для пружин поводка в два рази менше за жорсткості пружин базової мотузки. На першому знімку 5 з серії бачимо, що база має параболічну форму. Вантаж вже падає, але ще не відбувся момент натягу поводка. На знімку 6 бачимо ту саму форму, яку ми прогнозували побачити і схематично зобразили її на Рис. 9, спираючись на відеозаписи випробувань. На останніх двох знімках з серії 6-7 спостерігаємо форму мотузки вже після натягу поводка. Видно, що спочатку відбувається натяг пружин поводка, а потім поводок змушує тягнути за собою центральну точку бази, а вона в свою чергу тягне за собою донизу деякий фрагмент мотузки зліва і справа (тобто тягне деякий свій окіл).

З власних спостережень за поведінкою побудованої моделі через деякий достатньо великий проміжок часу зазначимо, що коли відбувається поступове

затухання коливань можна більш явно помітити синусоїдну форму поперечних коливань базової мотузки. Особливо це помітно у моменти часу, коли вантаж рухається з найнижчого положення вверх і відповідно пружинки мотузки-поводка повертають енергію догори.

Нижче наведемо лістинги з фрагментами коду програми.

4. Лістинги програми

Лістинг 4.1. Імпорт необхідних нам бібліотек і модулів.

```
import numpy as np
import pymunk as pm
import pygame as pg
import pymunk.pygame_util
import datetime
import time
```

Лістинг 4.2. Початкові налаштування програми. Задання частоти оновлення кадрів у секунду. Ініціалізація бібліотеки Pygame і об’єкту простору Pymunk.

```
FPS = 60
pg.init()
screen = pg.display.set_mode((800, 650))
clock = pg.time.Clock()
space = pm.Space()
space.gravity = (0,100)
do_snapshots = False  #Змінна, що визначає необхідність робити знімки вікна
interval = 1.05  # Інтервал між знімками у секундах
num_snapshots = 12  # Кількість знімків
snapshot_delay = 2.2 #Затримка у секундах початку роблення знімків
```

Лістинг 4.3. Задання гіперпараметрів системи.

```
#Гіперпараметри
base_points_number = 30 #12 #24 #18 #кількість точок бази без центральної точки
loaded_rope_points_number = 3    #кількість точок поводка без центральної точки та без точки-вантажа
x_left_coord, x_right_coord = 30, 770  #координати фіксованих точок по осі x
y_coord = 120                    #координата фіксованих точок і точок бази по y
x_center_point_coord =  (770 + 30) / 2  #координата центральної точки бази
center_weight_distance = 150    #відстань від центральної точки бази до точки-вантажа без деформації
base_rope_mass = 8               #маса всієї мотузки
base_rope_stiffness =  7  #3.8 #7       #коефіцієнт жорсткості всієї базової мотузки
#base_spring_stiffness, base_spring_damping = 1.5, 0.05#1.5, 0.0015 #1.5, 0.02 #пружність і коефіцієнт в'язкості для пружинок бази
base_spring_stiffness,base_spring_damping =
base_rope_stiffness * base_points_number, 0.05
loaded_rope_stiffness,loaded_rope_damping=base_spring_stiffness/2,
base_spring_damping/2
delta_len = 150     #розмір подовження пружин бази у початковий момент часу
#base_point_mass, base_point_moment = 0.001, 1     #маса і момент інерції точок бази
base_point_mass, base_point_moment = base_rope_mass / base_points_number, 1
static_edges_radius = 10                    #радіус статичних точок
loaded_rope_point_mass, loaded_rope_point_moment = 1, 1         #маса і момент інерції точок поводка
weight_point_mass, weight_point_moment = 30, 5   #маса і момент інерції точки-вантажа
static_point_mass, static_point_moment = 1, 0
```

moving_points_radius = 7 #радіус рухомих точок

x_bias, y_bias = 0, 0 #зміщення по осі x і по осі y точки-вантажа відносно центральної точки

Лістинг 4.4. Функція, що створює нерухомі крайні точки і додає їх до простору space, а також створення конкретних точок edges (після функції).

```
def create_static_edges(space):
    #Створення нерухомих крайніх точок
    left_edge_body = pm.Body(static_point_mass, static_point_moment, body_type=pm.Body.STATIC)
    right_edge_body = pm.Body(static_point_mass, static_point_moment, body_type=pm.Body.STATIC)
    left_edge_body.position, right_edge_body.position = (x_left_coord, y_coord), (x_right_coord, y_coord)
    left_shape, right_shape = pm.Circle(left_edge_body, static_edges_radius), pm.Circle(right_edge_body, static_edges_radius)
    space.add(left_edge_body, right_edge_body, left_shape, right_shape)
    return {'bodies': (left_edge_body, right_edge_body), 'shapes': (left_shape, right_shape)}

edges = create_static_edges(space)
```

Функція з лістингу 4.4. повертає словник, що складається з двох полів: об’єкти-тіла (тобто фізичні тіла) та геометричні форми цих тіл.

Лістинг 4.5 Функція малювання крайніх точок.

```
def draw_edges(edges):
    #Функція, для малювання крайніх точок
```

```
    for edge in edges['shapes']:
        pos_tuple = (int(edge.body.position.x),int(edge.body.position.y))
        pg.draw.circle(screen, (0, 0, 50), pos_tuple, static_edges_radius)
```

Лістинг 4.6. Функція створення і додавання до простору усіх рухомих точок, а також пружин, що їх з'єднують.

```
def add_moving_circles(space, edges):
    #Функція додавання рухомих точок посередині
        center_point = pm.Body(base_point_mass, base_point_moment, body_type = pm.Body.DYNAMIC)
        center_point.position = pm.Vec2d(x_center_point_coord, y_coord)
        shape = pm.Circle(center_point, moving_points_radius)

       weight_point = pm.Body(weight_point_mass, weight_point_moment, body_type = pm.Body.DYNAMIC)
     weight_point.position = pm.Vec2d(x_center_point_coord + x_bias, y_coord - center_weight_distance - y_bias)
     weight_point_shape = pm.Circle(weight_point, moving_points_radius)
    center_point.velocity, weight_point.velocity = pm.Vec2d(0, 0), pymunk.Vec2d(0,0)

     point_shapes, point_bodies, springs = [],[],[]
          x_base_point_coords = np.linspace(edges['bodies'][0].position[0], edges['bodies'][1].position[0], base_points_number + 3)
           y_loaded_rope_coords = np.linspace(center_point.position.y, weight_point.position.y, loaded_rope_points_number + 2)[1:]

#stretched_len_base_spring = abs(edges['bodies'][0].position[0] - x_base_point_coords[1])
#rest_len_loaded_rope_spring = abs(y_coord - y_loaded_rope_coords[1])
```

```
stretched_len_base_spring=edges['bodies'][0].position.get_distance((x_base_point_coords[1], y_coord))
delta = delta_len if 0 <= delta_len < stretched_len_base_spring else 0.5 * stretched_len_base_spring
rest_len_base_spring = stretched_len_base_spring -  delta
rest_len_loaded_rope_spring=center_point.position.get_distance((x_center_point_coord, y_loaded_rope_coords[0]))


    for i in range(len(x_base_point_coords)):
        #цикл додавання точок і пружин бази
        if i == 0:
            continue
        elif i == len(x_base_point_coords) - 1:
            spring = pm.constraints.DampedSpring(point_bodies[-1], edges['bodies'][1], anchor_a = (0,0), anchor_b = (0,0), rest_length = rest_len_base_spring, stiffness = base_spring_stiffness, damping = base_spring_damping)

        else:
            left_point = edges['bodies'][0] if i == 1 else point_bodies[-1]

            if i == int(len(x_base_point_coords) / 2):    #if i == index of the center point x_coord
                point = center_point
                point_shape = shape

            else:
                point = pm.Body(base_point_mass, base_point_moment, body_type = pm.Body.DYNAMIC)
            point.position = pm.Vec2d(x_base_point_coords[i], y_coord)
```

```
            point_shape = pm.Circle(point, moving_points_radius)


            spring = pm.constraints.DampedSpring(left_point, point, anchor_a = (0,0), anchor_b = (0,0), rest_length = rest_len_base_spring, stiffness = base_spring_stiffness, damping = base_spring_damping)


            point.velocity = pm.Vec2d(0, 0)
            point_shapes.append(point_shape)
            point_bodies.append(point)
            space.add(point, point_shape)


        spring.activate_bodies()
        springs.append(spring)
        space.add(spring)


    loaded_rope_point_shapes, loaded_rope_point_bodies, loaded_rope_point_springs = [],[],[]
    for i in range(len(y_loaded_rope_coords)):
        #Loop for adding points of loaded spring
        if i == len(y_loaded_rope_coords) - 1:
            a_point = center_point if i == 0 else loaded_rope_point_bodies[i - 1]
            point, point_shape  = weight_point, weight_point_shape

        else:
            a_point = center_point if i == 0 else loaded_rope_point_bodies[i - 1]
```

```
            point = pm.Body(loaded_rope_point_mass, loaded_rope_point_moment, body_type = pm.Body.DYNAMIC)
            point.position = pm.Vec2d(x_center_point_coord, y_loaded_rope_coords[i])
            point_shape = pm.Circle(point, moving_points_radius)

            spring = pm.constraints.DampedSpring(a_point, point, anchor_a = (0,0), anchor_b = (0,0), rest_length = rest_len_loaded_rope_spring,
                                                 stiffness = loaded_rope_stiffness,
                                                 damping = loaded_rope_damping)


        loaded_rope_point_shapes.append(point_shape)
        loaded_rope_point_bodies.append(point)
        space.add(point, point_shape)

        loaded_rope_point_springs.append(spring)
        spring.activate_bodies()
        space.add(spring)


    return {'base_shapes': point_shapes, 'base_springs': springs, 'loaded_rope_shapes': loaded_rope_point_shapes, 'loaded_rope_springs': loaded_rope_point_springs}
```

Ця функція повертає словник з чотирма ключами 'base_shapes', 'base_springs', 'loaded_rope_shapes', 'loaded_rope_springs', що містять дані про об’єкти тіл точок і пружин, а також дані про їх форми.

Лістинг 4.7. Функція малювання рухомих точок і пружин між ними.

```
def draw_circles(circles):
```

```
    #Drawing all objects using pygame interface
    for base_shape in circles['base_shapes']:
        pos_tuple = (base_shape.body.position.x, base_shape.body.position.y)
        pg.draw.circle(screen, (0,0,50), pos_tuple, moving_points_radius)

    for i, lr_shape in enumerate(circles['loaded_rope_shapes']):
        pos_tuple = (lr_shape.body.position.x, lr_shape.body.position.y)
        colors = (0, 0, 250) if i == len(circles['loaded_rope_shapes']) - 1 else (0,0,50)
        pg.draw.circle(screen, colors, pos_tuple, moving_points_radius)

    for base_spring in circles['base_springs']:
        pg.draw.line(screen, (0,0,0), (base_spring.a.position.x, base_spring.a.position.y),
                (base_spring.b.position.x, base_spring.b.position.y), 1)

    for lr_spring in circles['loaded_rope_springs']:
        pg.draw.line(screen, (0,0,0), (lr_spring.a.position.x, lr_spring.a.position.y),
                (lr_spring.b.position.x, lr_spring.b.position.y),  1)
```

У цій функції програма послідовно виконує 4 цикли, в кожному з яких ітерується по відповідним об’єктам звертаючись до потрібних полів словника, який повертає функція з попереднього лістингу. У перших двох циклах малюються рухомі точки бази і поводка. При бажанні можна вручну налаштувати їх колір і радіус. У другій парі циклів малюються пружини, які на екрані відображаються лініями. Можна налаштувати її колір у форматі RGB і товщину у пікселях.

Лістинг 4.8. Основний цикл програми.

```
circles = add_moving_circles(space, edges)
```

```
snapshot_timer = 0
snapshot_counter = 0

loop_start_time = time.time()
while True:
    for event in pg.event.get():
        if event.type == pg.QUIT:
            pg.quit()

    screen.fill((255,255,255))
    draw_edges(edges)
    draw_circles(circles)
    space.step(1/FPS)
    if do_snapshots and time.time() > loop_start_time + snapshot_delay:
        snapshot_timer += clock.get_time()
        if snapshot_timer >= interval * 1000 and snapshot_counter < num_snapshots:
     snapshot_datetime = datetime.datetime.now().strftime("%d%m%Y_%H%M%S")
  pg.image.save(screen, f"screenshot_{snapshot_counter}_{snapshot_datetime}.png")
            snapshot_counter += 1
            snapshot_timer = 0
    pg.display.update()
    clock.tick(FPS)
```

У цьому фрагменті коду відбувається оновлення вікна програми і послідовний виклик функцій, що малюють об’єкти на екрані. Програма працюватиме до тих пір, поки не буде закрите вікно Pygame. Можна за бажанням робити знімки екрану у певні моменти часу, а також налаштувати затримку у секундах після якої починати їх робити. Скріншоти зберігаються у папці, де розташований файл програми.

Висновки

У роботі розглянута модель, що складається з двох з’єднаних мотузок і вантажу. Виведені диференціальні рівняння для опису руху вантажа. Розглянута модель може бути модифікована шляхом паралельного під’єднання демпферів до кожної з пружинок обох мотузок. Таким чином можна буде також враховати в’язку складову у рівняннях руху (у цьому випадку у рівняннях буде також доданок з першою похідною від координат помноженою на коефіцієнт в’язкості). Варто зазначити деякі особливості цієї моделі.

По-перше, це те, що жорсткість пружин бази має бути більшою у 1.5-2 рази за жорсткість пружин мотузки-поводка. Ця умова є необхідною, тому що база повинна сповільнювати дуже швидкий рух вантажа, а також зменшувати амплітуду коливань.

По-друге, деяким спрощенням у моделі є те, що точка C є фіксованою, і не рухається вздовж базової мотузки. В житті зазвичай ця точка закріплена на ролику, яких рухається по базовій мотузці. Також симетричність відносно вертикальної осі дає нам змогу взагалі кажучи розглядати рівняння тільки точок однієї з половин базової мотузки. Зміщення ж цієї точки відносно центра означає, що відповідні кути α_i між пружинами і горизонтальним напрямком будуть різними для симетричних точок.

По-третє, обмеження на рух мотузки-поводка дає нам змогу не додавати до системи рівняння, які містили б x-координати її точок.

Також ми вважаємо, що початковий момент часу у задачі Коші є моментом, коли мотузка-поводок ще не натягнута, але має натуральну довжину. Це пояснюється тим, що найцікавіша поведінка, яка і являє собою саму суть дослідження моделі, виникає якраз після цього моменту часу. Ми припускаємо, що ця модель має пояснювати форму мотузкової системи у моменти навантаження, яка формується внаслідок розповсюдження поздовжніх і поперечних хвиль.

Швидкість руху останніх є значно меншою, тому вони впливають на форму з запізненням, порівняно з поздовжніми хвилями.

Нам вдалося побудувати комп'ютерну модель за допомогою мови програмування Python і допоміжних бібліотек. Ми провели багато експериментів з геометрією системи і зміною гіперпараметрів. Отримали знімки форми мотузкової системи у різні моменти часу.

Вивчення цієї і подібних моделей дозволяє робити оцінки максимальних навантажень на систему, що виникають внаслідок дій сил натягу. Особливо важливе значення мають точки кріплення A, B, D, оскільки вони мають бути надійними.

Список використаних джерел

1. Ulrich Leuthausser. The physics of a climbing rope under a heavy dynamic load. *Journal of sports engineering and technology*. 2017, Vol. 231(2) P. 125-135.
2. Ulrich Leuthausser. Physics of a climbing ropes: impact forces, fall factors and rope drag, part 2, Version 3 (12.7.2016).
3. R.M. Christensen, Theory of Viscoelasticity An Introduction, 2nd Edition, Academic Press, 2012.
4. David Morin. Introduction to Classical Mechanics With Problems and Solutions, Cambridge University Press, 2008.
5. John R. Taylor. Classical Mechanics, University Science Books, 2005.
6. Richard Haberman. Elementary Applied Partial Differential Equations with Fourier Series and Boundary Value Problems, Second Edition, Prentice Hall College Div, Englewood Cliffs, New Jersey, 1987.
7. https://www.python.org/
8. https://www.pymunk.org/en/latest/#
9. https://www.pygame.org/news