НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Приладобудівний факультет
Кафедра приладів і систем орієнтації і навігації

До захисту допущено:
Завідувач кафедри
_____ Надія БУРАУ
«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою «*Комп'ютерно - інтегровані технології та системи навігації* і *керування*»
спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології»
на тему: «Автоматизація структурного та кінематичного аналізу плоского важільного механізму»

Виконав:
студент IV курсу, групи ПГ-71
Русняк Давид Павлович _____

Керівник:
Доцент, к.т.н,
Цибульник С.О. _____

Рецензент:
Доц. к.т.н.,
Нечай С.О. _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2021 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут/факультет Приладобудівний
(повна назва)

Кафедра приладів і систем орієнтації і навігації
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 151 – «Автоматизація та комп'ютерно-інтегровані технології»
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри ПСОН
_____ Н.І. Бурау
(підпис) (ініціали, прізвище)
«___»_____20__ р.

ЗАВДАННЯ
на дипломний проект (роботу) студенту

Русняк Давид Павлович
(прізвище, ім'я, по батькові)

1. Тема проекту Автоматизація структурного та кінематичного аналізу плоского важільного механізму

керівник проекту (роботи) _____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «25» травня 2021 р. № 1180с

2. Строк подання студентом проекту (роботи) 12.06.2020 р.

3. Вихідні дані до проекту (роботи)

4. Зміст (дипломної роботи) пояснювальної записки (перелік завдань, які потрібно розробити) Вступ.

5. Перелік графічного (ілюстративного) матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо) _____
_____
_____
_____

6. Консультанти розділів проекту (роботи)*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 18.05.2020_____

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз завдання до дипломного проекту	18.05.2020 – 20.05.2020	
2	Огляд існуючих рішень	21.05.2020 – 23.05.2020	
3	Вибір та обґрунтування елементів	24.05.2020 – 28.05.2020	
4	Проектування Програмного забезпечення	29.05.2020 – 02.06.2020	
5	Програмування	03.06.2020 – 04.06.2020	
6	Аналіз працездатності	04.06.2020 - 05.06.2020	
7	Оформлення пояснювальної записки та графічних документів	06.06.2020 – 07.06.2020	

Студент _____ Русняк Д.П.
(підпис) (ініціали, прізвище)

Керівник проекту (роботи) _____ Цибульник С.О.
(підпис) (ініціали, прізвище)

* Консультантом не може бути зазначено керівника дипломного проекту (роботи)

АНОТАЦІЯ

У ході даної дипломної роботи було зроблено огляд технологій та методів аналізу плоских важільних механізмів, розглянуто теоретичні відомості, види , принципи побудови та роботи механізмів.

За допомогою об’єктно-орієнтованої мови програмування – Processing, було створено програмне забезпечення для забезпечення автоматизації процесів структурного та кінематичного аналізу плоских важільних механізмів.

Проведено випробовування працездатності системи для перевірки її ефективності.

Пояснювальна записка складає 61 сторінку та містить 24 рисунки, 2 таблиці.

ABSTRACT

During this graduate work, an inspection of technologies and methods of analysis of flat lever mechanisms was made, theoretical information, types, principles of construction and operation of mechanisms were considered.

Using the object-oriented programming language - Processing, software was created to automate the processes of structural and kinematic analysis of flat lever mechanisms.

The system was tested to verify its effectiveness.

СПИСОК УМОВНИХ СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

КЛ – кінематичний ланцюг

ПМ – плоский механізм

КП – кінематична пара

ТММ – теорія механізмів і машин

ЗМІСТ

ВСТУП

Теорія механізмів і машин є однією з основоположних дисциплін в інженерії , яка досліджує властивості механізмів і машин, основні методи і алгоритми синтезу і аналізу.

В сучасному світі галузь машинобудування є основою галуззю промислово розвинутої країни – це є основою технічного прогресу всіх галузей народного господарства. Сам прогрес визначається досконалістю створюємих машин. Саме з цієї причини інженери повинні мати глибокі теоретичні знання та досвід, для вміння успішного використання та забезпечення швидкого прогресу складної техніки.

Основною ціллю даної дипломної роботи поставлено автоматизація та створення програмного забезпечення для швидшого , варіативного і якіснішого аналізу механізмів і машин.

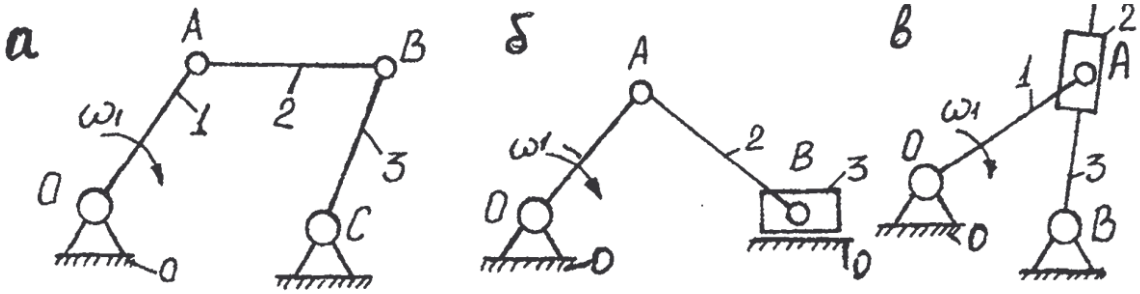
РОЗДІЛ 1. ОГЛЯДОВА ЧАСТИНА

1.1 Загальні відомості, принцип роботи, приклади механізмів

Механізм- це пристрій для перетворення механічного руху твердих тіл[1].Всі механізми, що складаються із твердих тіл, діляться на:

1) Важільні(рис 1.1)
2) Кулачкові(рис 1.2)
3) Фрикційні, що працюють за рахунок тертя(рис 1.3)
4) Зубчасті, що складаються із зубчастих коліс(рис 1.4)
5) З гнучкими ланками(рис 1.5)
6) Гідравлічні і пневматичні, робочим тілом яких є рідина, або повітря
7) Лічильні(планіметри, інтегратори, аналізатори і ін.)[2]

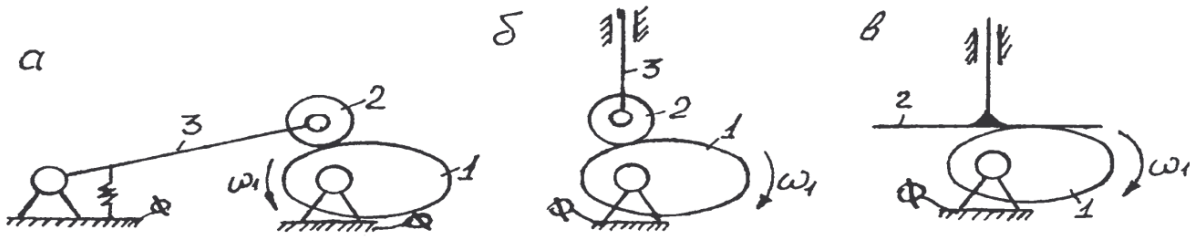
Важільні механізми



а) кривошипно-коромисловий; б) кривошипно-повзунний в) кулісний

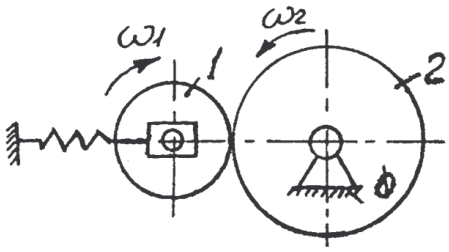
Рис 1.1

Кулачкові Механізми

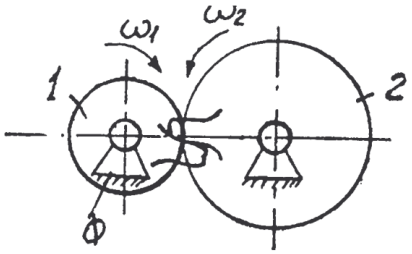


а) з коромисловим штовхачем; б) з роликовим штовхачем в) з тарілчастим штовхачем

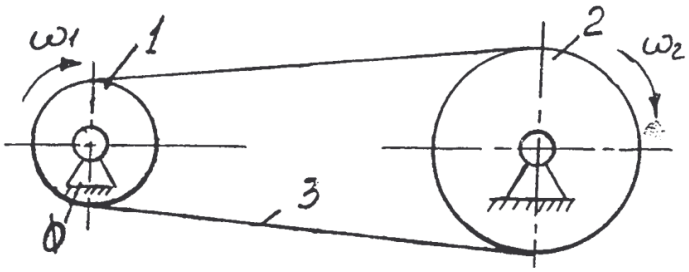
Рис 1.2



Фрикційний механізм Рис 1.3



Зубчастий механізм Рис 1.4



Механізм з гнучкою ланкою Рис 1.5

Будь-який механізм складається із окремих деталей. Виріб, що виконаний із однорідного матеріалу, називається деталлю , які між собою можуть бути з’єднанні рухомо або нерухомо.

Деталь, або нерухоме з’єднання декількох деталей, що несе елемент и кінематичних пар, називається ланкою.

Місце рухомого з’єднання двох ланок називається кінематичною парою.

Всякий механізм має нерухому ланку, або ланку, що сприймається за нерухому, яка називається стояком (0- рис 1.1).

Із рухомих ділянок виділяють вхідні та вихідні (ведучі та ведені).

Вхідною ланкою (ведучою) називається ланка, закон руху і сили якій задається (1 -рис. 1.5).

Вихідною (веденою) називається ланка, від якої отримуються потрібні рухи і сили для робочого органу механізму (3 —рис. 1.5).

Останні ланки називаються з’єднувальними, або проміжними. В більшості випадків в механізмах одна вхідна 1 одна вихідна ланка. Вхідна ланка отримує рух від двигуна, а вихідна з'єднується з робочим органом машини.

Залежно від характеру руху відносно стояка інші ланки називають:

Кривошипом - обертову ланку, яка може здійснювати повний оберт навколо нерухомої осі (1- рис. 1.1);

Шатуном - ланку, що з’єднана обертальними кінематичними парами тільки з рухомими ланками (2-рис. 1.1, а, 6):

Повзуном - ланку, яка утворює поступальну пару одною ланкою і обертальну з другою (3-рис. 1.1, б) ;

Коромислом - обертову ланку, що здійснює неповний оберт навколо нерухомої осі ((3-рис. 1.1, а);

Кулісою - обертову ланку, яка є напрямною повзуна (3- рис. 1.1, в):

Ковзаючим каменем - ланку, шо здійснює прямолінійно-поступальний рух вздовж рухомої напрямної (2-рис. 1.1, в):

Напрямною -ланку, що обмежує рух ковзаючого каменю (3-рис. 1.1, в). або повзуна (0-рис. 1.1, б);

При зображенні механізму на кресленні, розрізняють його структурну схему із застосуванням умовних позначень ланок і кінематичних пар (без дотримання масштабу) і кінематичну схему, яка є його кінематичною моделлю (із дотриманням масштабу).

Структурна схема містить загальну інформацію про механізм: про кількість ланок та кінематичних пар, послідовність, способи 3'єднання ланок та види можливих рухів (рис. 1.1)

Кінематична схема механізму будується у вибраному масштабі з точним дотриманням всіх розмірів і форм, від яких залежить рух тієї чи іншої ланки. На кінематичній схемі повинно бути вказане все, що є необхідним для вивчення руху[2].

1.2 Важільні механізми

Важільні механізми - механізми, що складаються з жорстких ланок з'єднаних кінематичними парами, які здатні лише на обертальний або поступальний рух. На рис. 1.6 показаний двохланковий важільний механізм, на рис. 1.7 кривошипно-шатунний механізм, на рис. 1.8- чотирьохланковий кулісний механізм, на рис. 1.9- чотирьохланковий важільний механізм. [3]

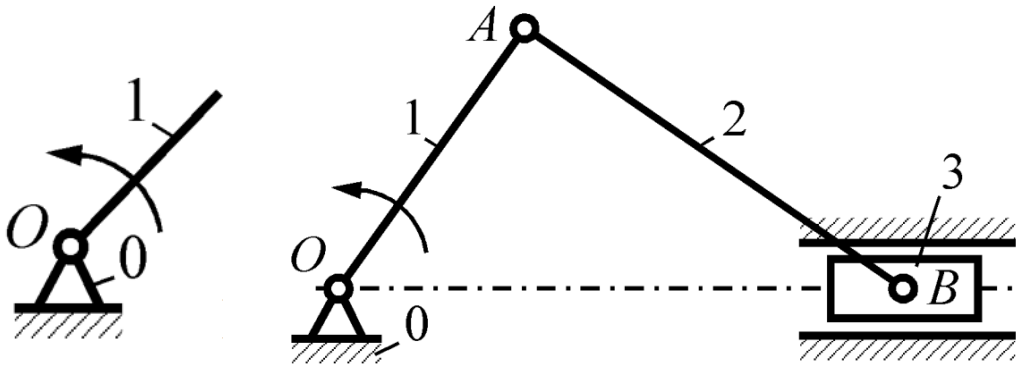


Рис. 1.6

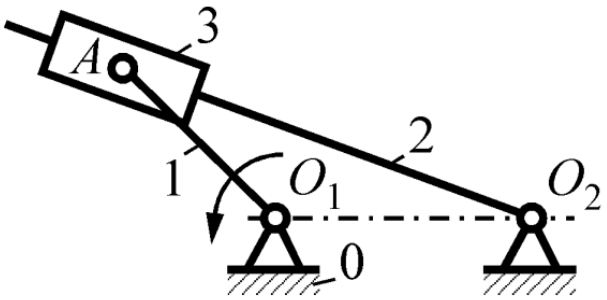


Рис. 1.7

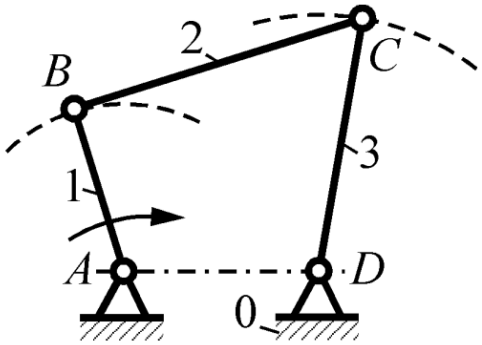


Рис. 1.8

Рис. 1.9

1.3 Рухливість кінематичного ланцюга

Число ступенів свободи КЛ щодо однієї з ланок називають ступенем її рухливості . Для визначення ступеня рухливості кінематичного ланцюга W необхідно із загального числа ступенів свободи всіх її рухомих ланок відняти число зв'язків, що накладаються на відносний рух ланок кінематичними парами, які пов'язують ланки. Нехай n - число рухомих ланок просторової кінематичного ланцюга; P_i - Число кінематичних пар i-го класу ($i = 1, \ldots .5$) Тоді 6n - загальне число ступенів свободи n ланок ланцюга, якщо вважати їх не пов'язаними між собою, а iP_i - загальне число зв'язків, накладених на ланки механізму кінематичними парами i-го класу.[4]

Ступінь рухливості КЛ можна визначити виразом:

$$W = 6n - \sum_{i=1}^{5} iP_i \qquad (1)$$

Розгорнувши суму у виразі (1), можна отримати формулу просторової КЛ загального вигляду для просторових механізмів(формулу Сомова - Малишева):

$$W = 6n - 5P_5 - 4P_4 - 3P_3 - 2P_2 - P_1 \qquad (2)$$

Та у випадку для плоских механізмів:

$$W = 3n - 2P_5 - P_4 \qquad (3)$$

1.3 Основний принцип створення механізмів

Вперше принципом утворення механізмів був чітко сформульований У 1914 р. російським вченим Л.В.Ассуром. Головна мета цього принципу полягає в тому, що будь-які механізми, в яких немає зайвих зв'язків і місцевих рухомостей, утворєються з КЛ який має якусь рухомість та КЛ з нульовою рухомістю.

Механізми першого класу утворюються з двох ланок, одна з котрих повинна бути нерухомою.

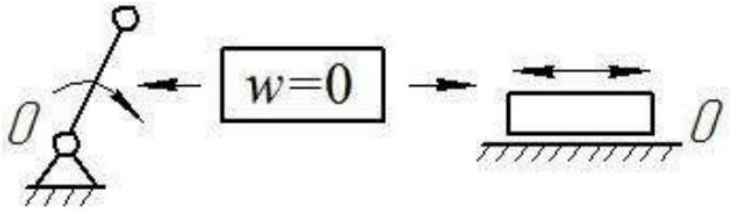


Рис 1.10 Приклади механізмів першого класу

Проте механізми першого класу можуть бути і самостійними механізмами, такими як ротор електродвигуна, або паровий молот.

Група Ассура

Відкритий КЛ, ступінь свободи якого дорівнює нулю і який не можна розділити на більш прості КЛ з нульовою ступінню вільності називаються *Групою Ассура*.

Для таких кінематичних груп відповідає рівняння:

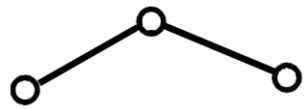
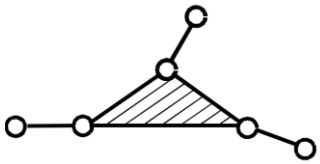
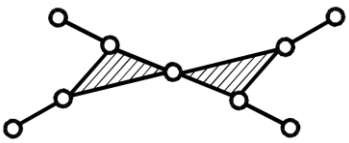
$$3n = 2p_5 \qquad (4)$$

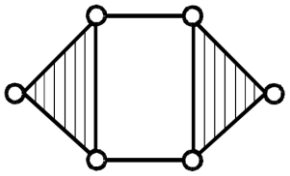
В.Ассур запропонував для найрозповсюдженіших сучасних механізмів ,структурні групи поділити на класи і порядки наведені в табл.1.1

За класифікацією Ассура до груп п I-го і II-го класів відносяться ті групи, що не мають змінних контурів. Різниця полягає в тому що до груп II-го класу входять базисні ланки, які з'єднуються тільки з іншими базисними ланками. Групи III-го класу містять один змінний контур. Групи IV-го класу – два змінних контури. В даній класифікації поряд групи визначається за допомогою числа повідків.

Також на сьогоднішній день існує класифікація за Артоболевським, яку використовують в переважному числу випадків. За цією класифікацією клас групи визначається за допомогою кількості шарнірів в найскладнішому замкненому контурі. Порядок групи визначається кількістю шарнірів, якими група приєднується до механізму.[1]

Таблиця 1.1 Найтиповіші групи Ассура та їх класифікація

Вид групи	**Класифікація**	
	За Ассуром	**За Артоболевським**
	I-й клас 2-й порядок	II-й клас 2-й порядок
	I-й клас 3-й порядок	III-й клас 3-й порядок
	I-й клас 4-й порядок	III-й клас 4-й порядок

	III-й клас 0-й порядок	IV-й клас 2-й порядок

1.4 Структурний аналіз та класифікація механізму

Структурний аналіз- це дослідження структури механізму, тобто визначення числа ланок і структурних груп, числа і типу кінематичних пар, кількості і виду кінематичних ланок, надлишкових і пасивних зв'язків. [5]

Структурна класифікація механізмів - один з найраціональніших класифікацій ПМ, основи якого буди закладені Л. В. Асууром і в подальшому розвинуто І. І. Артоболевським, В.В. Добровольським та ін. вченими.

Основа структурної класифікації механізмів являє собою основний принцип створення механізмів. Будь-який механізм можна створити за допомогою додавання до механізму I класу структурних груп, якщо задовольняються умови залежності (2) або (3).

Для пар IV класу,які можна замінити парами V класу, тому що всі кінематичні пари входять до складу ПМ, переписавши залежність (3) отримуємо:

$$3n - 2p_5 = 0, \qquad (5)$$

Звідки

$$p_5 = \frac{3}{2}n. \qquad (6)$$

Для того щоб задовольнити даний вираз , число ланок і пар має бути тільки цілим, а число ланок бути тільки парним, тому використовуються тільки такі числа ланок і КП які наведені в табл. 1.2.

n	2	4	6	8	…

p_5	3	6	9	12	…

Усі створенні групи за допомогою сполучення цих чисел, можна поділити на класи.

Структурні групи і механізми II класу

Як вже стало зрозуміло, найпростішою групою буде група, що утворена з двох ланок і трьох КП V класу (рис 1.11 а) (n=5,$p_5 = 3$). Таку групу називають структурна група II класу 2 порядку, або двоповідкова група.

Групи II класу бувають п’яти видів в залежності від поступових і обертових пар на їхнього взаємного положення(рис 1.11):

I вид II класу – група яка має дві ланки і три обертові пари;

II вид II класу – група в якій одну з крайніх обертових пар замінити на поступальну(рис 1.11, б, в);

III вид II класу – група в якій середню обертову пару замінено на поступальну (рис 1.11, г, д);

IV вид II класу – група в якій обидві крайні обертові пари замінено на поступальні (рис 1.11, е, є);

V вид II класу – група в якій замінена крайня і середня обертові пари(рис 1.11, ж, з), якщо замінити всі обертові пари на поступальні, то отримаємо клиновий механізм.

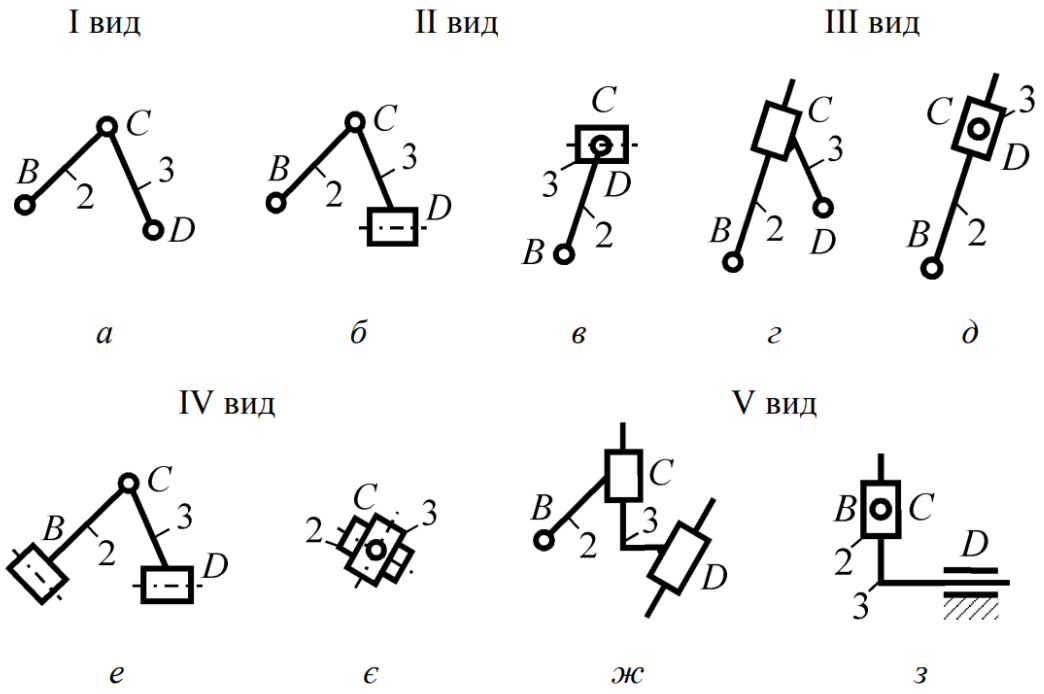


Рис 1.11 структурні групи II класу

На рис. 1.12 та 1.13 зображені приклади простих механізмів, де використовуються всі п’ять видів груп II класу.

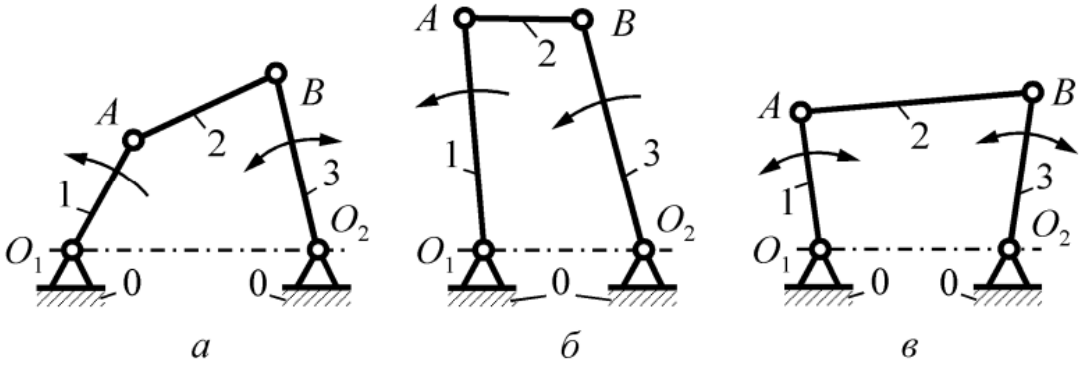


Рис.1.12 Види шарнірного чотириланкового механізму

На рис 1.12 зображено шарнірний чотириланковий механізм ABCD з ланками : 0- стояк; 1 – кривошип; 2 – шатун; 3 – коромисло. В залежності від розмірів ланок цей механізм може бути трьох видів: кривошипно-коромисловий (рис.1.12, а); двокривошипний (ланка 1 і 3 робить повний оберт, рис 1.12, б);двокоромисловий(ланка 1 і 3 здійснюють коливання, рис 1.12, в).

Механізми, в яких використовується група II класу II виду, називають кривошипно-повзунковим(Рис. 1.13, а); III виду – кривошипно-кулісний механізм (Рис. 1.13, б); IV виду – тангенсний механізм (Рис. 1.13, в); V виду – синусний механізм(Рис. 1.13, г).

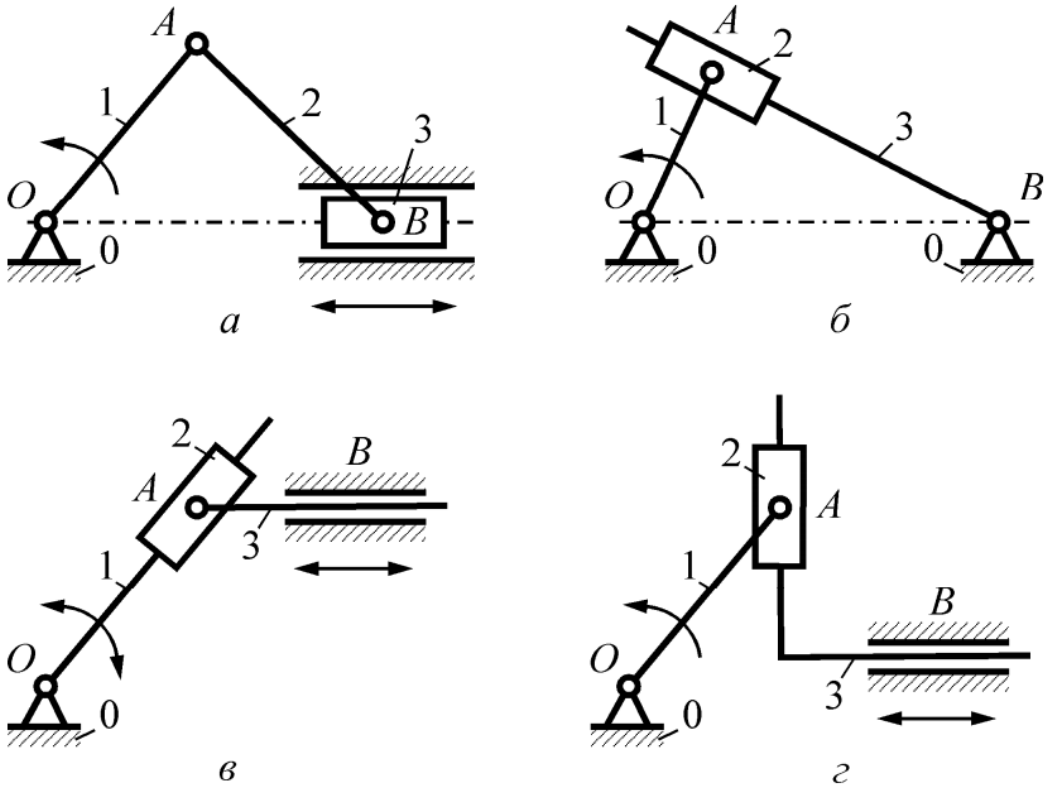


Рис 1.13 Види механізмів груп II-го класу

Порядок проведення структурного аналізу механізмів

Для проведення структурного аналізу механізму, треба пройти такі етапи як:

1. Визначення числа ступенів свободи механізму(або КЛ), у випадку наявності зайвих зв’язків або ступенів вільності, при структурному аналізі їх відкидають. Якщо є КП IV класу, то їх потрібно замінити на пари V класу з викресленням окремої структурної схеми замінного механізму.

2. Виділення початкових ланок – їх кількість залежить від числа ступенів вільності механізму(КЛ).

3. Розділення механізму на структурні групи, краще всього розпочинати із спроби роз’єднати від механізму групи II класу, перевіряючи при цьому число ступенів вільності W залишкової частини механізму, W повинно бути сталим. Групи відокремлюють до тих пір поки не залишиться механізм I класу. Якщо такий метод не дає результатів, слід переходити до спроб відокремлення груп III класу і тд.

4. Визначення класу і порядку структурних груп і класу механізму.

5. Запис формули побудови механізму.

1.5 Кінематичне дослідження механізмів

Кінематична схема механізму – умовне зображення механізму у масштабі $\mu_l(\frac{\text{м}}{\text{мм}})$,у якому дотримані лише ті розміри ланок,які впливають на кінематику механізму.

Тобто при дослідженні механізму розглядається рух ланок без урахування діючих на неї сил, інакше кажучи – рух ланок з геометричної точки зору.

Будь-який рух тіла можна охарактеризувати переміщенням, швидкістю і прискоренням, тому основні задачі кінематичного дослідження механізмів полягають:

1. Визначення положень ланок механізму, побудова траєкторії рухомих точок;

2. Визначення швидкостей окремих точок і ланок механізму;

3. Визначення прискорень окремих точок і ланок механізму;

В результаті чого отримуємо відповідність кінематичних параметрів (переміщень, швидкостей і прискорень), також отримують вихідні дані для

визначення сил і моментів (сил інерції, моментів сил інерції), кінетичної енергії та потужності механізму.[3]

Більшість механізмів і машин рухаються циклічно, це означає що через певний період часу, механізм повертається у початкове положення. Тому для кінематичного дослідження достатньо одного періоду роботи механізму. При цьому необхідно мати розміри всіх ланок і закон руху початкової ланки.

Існує чотири методи кінематичного дослідження механізмів:

- Графічний – дає змогу встановити картину взаємо-розташування ланок під час руху машини і усунути можливість їх співударів. Завдяки універсальності та наочності досліджень, графічний метод може використовуватись для механізмів будь-якої складності. Незважаючи на простоту реалізації, метод стає неефективним, коли необхідно провести великий об’єм побудов і розрахунків.
- Графоаналітичний – дозволяє визначити не тільки величину , але й напрямок кінематичних параметрів (швидкість, прискорення).
- Аналітичний - використовується для класу задач, коли ланки механізму повинні забезпечити рух за певним, наперед заданим законом. Даний метод дає можливість здійснювати багатоваріантні дослідження механізмів і тим самим вибирати такі схеми механізмів і розміри їхніх ланок, які забезпечують найкращі умови роботи.
- Експериментальний - дає найдостовірніші результати, які використовуються для одержання дійсних значень кінематичних параметрів механізму та перевірки результатів теоретичних досліджень. Зумовлено це тим, що для графічних і аналітичних методів розв'язання задач кінематики доводиться приймати ряд ідеалізованих припущень (ланки вважаються абсолютно жорсткими, у кінематичних парах відсутні зазори, всі ланки виготовлені абсолютно точно. головний вал машини обертається з постійною швидкістю тощо). У зв'язку з цим теоретичні залежності ідеалізованих механізмів можуть значно

відрізняються від дійсних. Крім цього, у машинах здебільшого використовуються механізми з пружними, гідравлічними, пневматичними зв'язками, теоретичні розрахунки яких також вимагають додаткової експериментальної перевірки.[3]

1.6 Побудова плану швидкостей важільного механізму

Метод подібності

Зобразимо довільне тіло К, яке здійснює плоский рух. Положення тіла визначається трьома точками А, В, С (рис. 1.14, а), які зв’язані з тілом і утворюють жорсткий трикутник АВС.

Нехай у нас є швидкості $\vec{v}_A,\ \vec{v}_B,\ \vec{v}_C$ відповідно точок А, В, С і положення миттєвого центра швидкостей Р. Вектор швидкості будь-якої точки направлений перпендикулярно до радіус-вектора, який з’єднує цю точку з точкою Р:

$$\vec{v}_A \perp \overrightarrow{PA},\ \vec{v}_B \perp \overrightarrow{PB},\vec{v}_C \perp \overrightarrow{PC} \qquad (7)$$

Швидкості точок пропорціональні радіусам-векторам

$$\frac{v_A}{PA} = \frac{v_B}{PB} = \frac{v_C}{PC} = \omega, \qquad (8)$$

ω – миттєва кутова швидкість тіла К.

Візьмемо тепер будь-яку довільну точку р на площині (рис. 1.14, б) і побудуємо в якомусь масштабі з даної точки вектори швидкостей точок А, В, С. З’єднавши точки a, b і c – кінці векторів швидкостей $\vec{v}_A,\ \vec{v}_B,\ \vec{v}_C$ – отримуємо план швидкостей тіла АВС.

Звідси можемо затверджувати що, планом швидкостей будь-якого тіла(ланки) є геометричне місце кінців векторів швидкостей крайніх точок тіла, відкладених з однієї довільної точки, що називається полюсом плану швидкостей.[3]

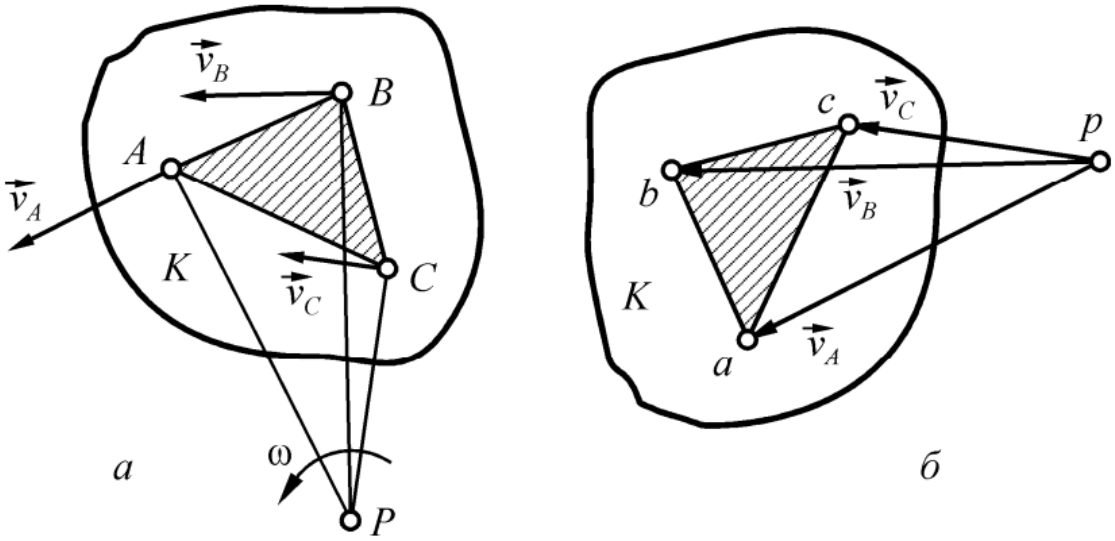


Рис 1.14

Оскільки відрізки pa, pb, pc перпендикулярні до радіусів відповідно PA, PB, PC і пропорційні їм, фігура pabc подібна фігурі PABC і повернута відносно неї на 90° в бік миттєвого обертання. Таким чином отримуємо теорему подібності для планів швидкостей: план швидкостей твердого тіла(ланки) подібний до тіла і повернутий відносно нього на 90° в бік миттєвого обертання.

Метод векторних рівнянь

Метод векторних рівнянь базується на теоремі кінематики про переміщення плоскої фігури: довільне переміщення плоскої фігури в її площині можна здійснити шляхом поступального переміщення разом з довільною точкою - полюсом, і обертального руху навколо полюса.[3]

До прикладу візьмемо побудову плану швидкостей кривошипно-повзункового механізму(Рис. 1.15) для якого задано кінематичну схему і закон руху кривошипа OA.

Для визначення точок механізму використаємо теорему про переміщення плоскої фігури - рух шатуна AB можна розкласти на: поступальний рух полюса – точки A, зі швидкістю $\vec{v}_A$, та обертальний рух навколо полюса зі швидкістю $\vec{v}_{BA}$. Справді, нехай спочатку всі точки шатуна

АВ рухаються поступально разом з полюсом зі швидкістю $\vec{v}_A$, при цьому вісь шатуна займе положенняA_1B_1'. Потім, зупинивши полюс — точку A_1, повернемо шатун АВ так, щоб точка B_1' потрапила на свою дійсну траєкторію х-х, тобто у точку B_1.

Отже, векторне рівняння плоского руху шатуна матиме вигляд

$$\vec{v}_B = \vec{v}_A + \vec{v}_{BA} \qquad (9)$$

Побудуємо план швидкостей: спочатку знайдемо швидкість точки А.

$$v_A = \omega_1 l_{OA} \qquad (10)$$

Де ω_1- кутова швидкість обертання кривошипа; l_{OA}- дійсна довжина кривошипа ОА.

Вектор $\vec{v}_A$ зображується відрізком *pa* (Рис. 1.15, б), який визначає модуль швидкості точки А у масштабі:

$$\mu_v = v_A/pa \qquad (11)$$

Через точку *a* проводиться паралельна лінія відносно вектора швидкості $\vec{v}_{BA}$, а з полюса *p* – паралельну лінію відносно руху повзуна В (||*x*-*x*). Точка, в якій перетнуться ці дві лінії визначать точку b – кінець векторів $\vec{v}_{BA}$ і $\vec{v}_B$.Відрізок ab визначає у масштабі модуль швидкості $\vec{v}_{BA}$, причому точка С, згідно з теоремою подібності, лежатиме на відрізку *ab*.

$$\frac{ac}{ab} = \frac{AC}{AB}, \qquad (12)$$

Склавши пропорцію (12), можемо отримати довжину відрізка:

$$ac = ab\frac{AC}{AB} \quad (13)$$

Відклавши відрізок ac на плані швидкостей і з'єднавши точку *c* з полюсом *p*, отримуємо швидкість точки С: $v_C = (pc)\mu_v$.

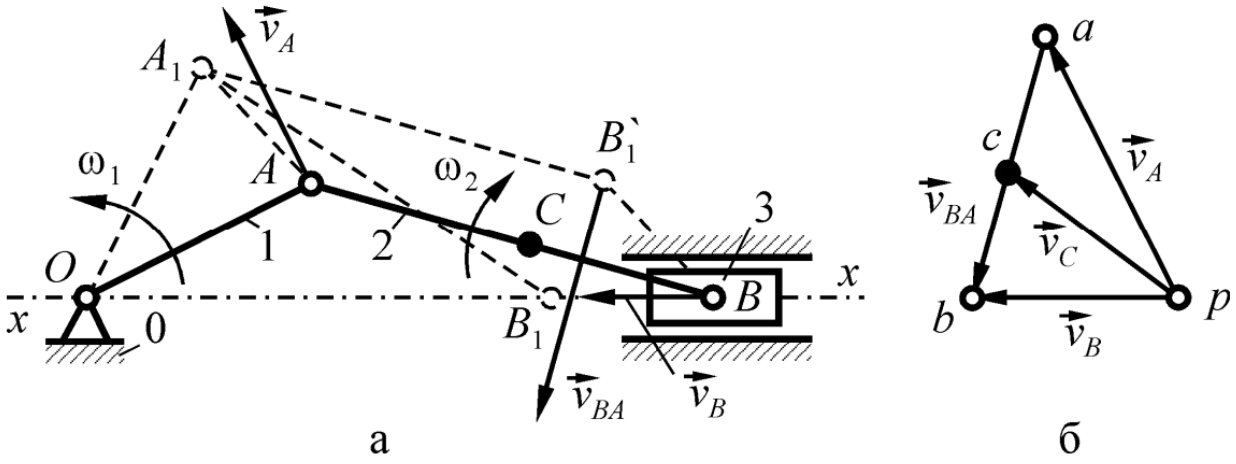


Рис. 1.15

Швидкість шатуна АВ знаходиться за формулою:

$$\omega_2 = \nu_{BA}/l_{AB},$$

Де $\nu_{BA} = (ab)\mu_\nu$. Кутова швидкість ω_2 відповідно до Рис.1.15 ,буде направлена за рухом годинникової стрілки.

1.7 Побудова плану прискорень методом векторних рівнянь

В ТММ, планом прискорення твердого тіла чи ланки називають геометричне місце кінців векторів прискорень крайніх його точок, відкладених з однієї довільної точки, що називається полюсом плану прискорень.[3]

Для плану прискорень теорема подібності звучить так: план прискорень будь-якого тіла(ланки) подібний до тіла і повернутий відносно нього на деякий невизначений кут.

Беручи до прикладу кривошипно-повзуновий механізм (Рис.1.16), розберемо приклад побудови планів прискорень, використовуючи векторне рівняння плоского руху шатуна(14), де вектор прискорення $\vec{a}_B$ направлений уздовж напрямної *x-x*.

$$\vec{a}_B = \vec{a}_A + \vec{a}_{BA}, \quad (14)$$

Прискорення точки А (Рис.1.16, а) являє собою геометричну суму нормального і дотичного прискорень:

$$\vec{a}_B = \vec{a}_A^n + \vec{a}_A^\tau. \qquad (15)$$

Нормальне прискорення $\vec{a}_A^n$ напрямлене вздовж кривошипа до його центра обертання – точки О, а дотичне $\vec{a}_A^\tau$ – перпендикулярно кривошипу і направлене в бік напрямку кутового прискорення ε_1 ланки 1.

$$a_A^n = \omega_1^2 l_{OA} = \frac{v_A^2}{l_{OA}},\ a_A^\tau = \frac{dv_A}{dt} = \varepsilon_1 l_{OA} \qquad (16)$$

Якщо початкова ланка рухається рівномірно($\omega_1 = const$), то $\varepsilon_1 = \frac{d\omega_1}{dt} = 0$, що означає $a_A^\tau = 0$, тобто:

$$\vec{a}_A = \vec{a}_A^n\,. \qquad (17)$$

Прискорення
$\vec{a}_{BA}$ розкладається на аналогічні до рівняння (15)дві складові:

$$\vec{a}_{BA} = \vec{a}_{BA}^n + \vec{a}_{BA}^\tau, \ (18)$$

Вектор нормального прискорення направлений уздовж шатуна від точки В до А, а дотичне – перпендикулярно до шатуна ($\vec{a}_{BA}^n \perp \vec{a}_{BA}^\tau$).

Прийнявши до урахування вирази (17)та (18) , рівняння (15) матиме вигляд:

$$\vec{a}_{BA} = \vec{a}_A + \vec{a}_{BA}^n + \vec{a}_{BA}^\tau \qquad (19)$$

Прийнявши точку π за полюс плану прискорень (Рис.1.16, б), потрібно відкласти вектор, зображаючи нормальне прискорення точки А, у вигляді відрізна πa. Масштабний коефіцієнт знаходиться за формулою:

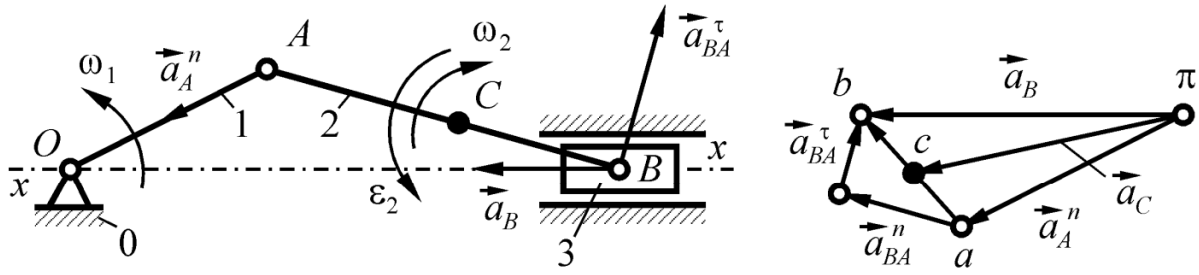
$$\mu_a = \frac{a_A}{\pi a},\ \text{м}/(\text{мм} \cdot \text{с}^2) \qquad (20)$$

Модуль прискорення a_{BA}^n визначається за формулою

$$a_{BA}^{n} = \omega_2^2 l_{AB} = \frac{v_{BA}^2}{l_{AB}} \qquad (21)$$

Прискорення a_{BA}^{n} зображено відрізком $an = a_{BA}^{n}/\mu_a$, який прикладений початком у точку a.Через його кінець проводимо паралельну дотичному прискоренню лінію, а через полюс π – лінію паралельну прискоренню точки B (||*x*-*x*). Точка перетину $\vec{a}_B$ і $\vec{a}_{BA}^{\tau}$ визначать точку b – кінець даних векторів.

З'єднавши точки a і b, знаходиться вектор повного прискорення $\vec{a}_{BA}$. З'єднавши точку π з серединою відрізка ab, знайдемо прискорення $\vec{a}_C (a_C = (\pi c)\mu_a)$.



а б

Рис.1.16

Модуль кутового прискорення шатуна $\varepsilon_2 = a_{\mathrm{BA}}^{\tau}/l_{AB}$. Відповідно до Рис.1.16, а , кутове прискорення ε_2 направлене проти руху годинникової стрілки.

РОЗДІЛ 2. ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Для зручнішого відображення та читання коду в програмному середовищі Processing, програма розділена на 9 частин :

- diplom – основний файл в якому знаходяться основні функції зображення та функціонування програми.
- Crossing – файл функцій що відповідають за розрахунок перетину точок елементів механізму
- Podstavka – клас на основі якого створюється нерухома основа
- Stoika – клас на основі якого створюється обертальна ланка
- Table – клас для відображення таблиці основних елементів
- Variables – файл для зберігання глобальних змінних та запобігання “засмічення” основного файлу
- Block - клас на основі якого створюється поступально рухома ланка
- Dotpoint – клас на основі якого створюються точки для з’єднання ланок
- Vector – Клас для розрахунку напрямку і швидкості векторів руху, швидкості і прискорення.

1. Огляд файлу dotpoint

$$float\ Xstart = 156;$$
$$float\ Ystart = 120;$$
$$float\ Radius\ = 12;$$

Змінні координат початкового положення точки та її радіус.

$$void\ drawPoint() \{$$
$$circle\ (Xstart, Ystart, Radius);$$
$$\}$$

Метод для створення кола з заданими параметрами.

Circle(“координата по осі X”,” координата по осі Y”,”радіус кола") – інтегрована функція для зображення кола.

$$public\ float\ getXstart() \{$$
$$return\ Xstart;$$
$$\}$$
$$public\ void\ setXstart(Float\ Xstart) \{$$
$$this.Xstart = Xstart;$$

$$
\begin{gathered}
\} \\
public\ float\ getYstart() \{ \\
return\ Ystart; \\
\} \\
public\ void\ setYstart(Float\ Ystart) \{ \\
this.Ystart = Ystart; \\
\} \\
public\ float\ getRadius() \{ \\
return\ Radius; \\
\} \\
public\ void\ setRadius(Float\ Radius) \{ \\
this.Radius = Radius; \\
\}
\end{gathered}
$$

Використання методу інкапсуляції створенням методів Get і Set для використання і зміни даних конкретно створених об’єктів

Результат виконання данного класу:

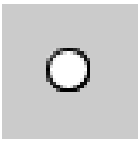


Рис.2.1

2. Огляд файлу Block

$$
\begin{gathered}
float\ Xstart = 155; \\
float\ Ystart = 200; \\
float\ widt = 25; \\
float\ heigh = 16; \\
float\ Radius = 12;
\end{gathered}
$$

Змінні координат початкового положення , розмірів та радіусу точки з’єднання

$$
\begin{gathered}
void\ drawBlock()\{ \\
rect(Xstart, Ystart, widt, heigh); \\
circle\ (Xstart, Ystart, Radius); \\
\}
\end{gathered}
$$

Метод для створення прямокутника та кола в центрі з заданими параметрами.

Rect(“координата по осі X”, ” координата по осі Y”, ”ширина”, ”висота”) - інтегрована функція для зображення прямокутника.

$$public\ float\ getXstart(\{$$
$$return\ Xstart;$$
$$\}$$
$$public\ void\ setXstart(Float\ Xstart)\{$$
$$this.Xstart = Xstart;$$
$$\}$$
$$public\ float\ getYstart(\{$$
$$return\ Ystart;$$
$$\}$$
$$public\ void\ setYstart(Float\ Ystart)\{$$
$$this.Ystart = Ystart;$$
$$\}$$
$$public\ float\ getRadius(\{$$
$$return\ Radius;$$
$$\}$$
$$public\ void\ setRadius(Float\ Radius)\{$$
$$this.Radius = Radius;$$
$$\}$$
$$\}$$

Аналогічні методи Get і Set.

Результат виконання данного классу:

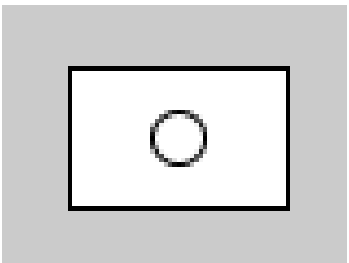


Рис.2.2

3. Огляд файлу Stoika

$$void\ drawStoika(\{$$

$$for\ (int\ i = 1;\ i <= 2;\ i++)\ \{$$

$$x\_vect\_end1 = Xstart + lengt * cos(Angl * i + rad);$$

$$y\_vect\_end1 = Ystart + lengt * sin(Angl * i + rad);$$

$$line(Xstart, Ystart, x\_vect\_end1, y\_vect\_end1);$$

$$if\ (i == 1)\ \{$$

$$x\_vect\_beg2 = x\_vect\_end1;$$

$$y\_vect\_beg2 = y\_vect\_end1;$$

$$\}$$

$$line(x\_vect\_beg2, y\_vect\_beg2, x\_vect\_end1, y\_vect\_end1);$$

$$\}$$

$$for\ (int\ i = 0;\ i < 50;\ i++)\ \{$$

$$if\ (\ lengt\ <\ hatch * i\ )\ \{$$

$$break;$$

$$\}$$

$$x\_strix\_beg = x\_vect\_end1 + (hatch * i * cos(rad));$$

$$y\_strix\_beg = y\_vect\_end1 + (hatch * i * sin(rad));$$

$$x\_strix\_end = x\_strix\_beg + (hatch * cos(hatchAngl + rad));$$

$$y\_strix\_end = y\_strix\_beg + (hatch * sin(hatchAngl + rad));$$

$$line\ (x\_strix\_beg, y\_strix\_beg, x\_strix\_end, y\_strix\_end);$$

$$\}$$

$$circle\ (Xstart, Ystart, Radius);$$

$$\}$$

drawStoika() – метод що відображає стояк. Перший цикл малює основний трикутник відносно початкових координат Xstart, Ystart, заданої довжини сторін lengt , та під заданим кутом $cos(Angl * i + rad)$ для координати X та $sin(Angl * i + rad)$ для Y. Другий цикл малює штрихування під стояком приймаючі задані значення довжини штрихування - $hatch$, куту нахилу штрихів - $hatchAngl$ та змінних початку і кінця координат - $x\_strix\_beg, y\_strix\_beg, x\_strix\_end, y\_strix\_end;$

Результат виконання данного классу:

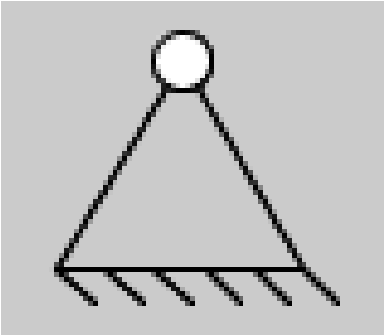


Рис. 2.3

4. Огляд файлу Podstavka

$$void\ drawPodstavka(\{$$

$$float\ rad\ =\ radians(angle);$$

$$float\ x\_vect\_end = Xstart + lengt * cos(rad);$$

$$float\ y\_vect\_end = Ystart + lengt * sin(rad);$$

$$line\ (Xstart, Ystart, x\_vect\_end, y\_vect\_end);$$

$$circle(Xstart, Ystart, Radius);$$

$$float\ x\_strix\_beg = 0, y\_strix\_beg = 0, x\_strix\_end = 0, y\_strix\_end = 0;$$

$$for\ (int\ i = 0;\ i < 50;\ i++)\ \{$$

$$if\ (\ lengt\ <\ hatch * i\ )\ \{$$

$$break;$$

$$\}$$

$$x\_strix\_beg = Xstart + (hatch * i * cos(rad));$$

$$y\_strix\_beg = Ystart + (hatch * i * sin(rad));$$

$$x\_strix\_end = x\_strix\_beg + (hatch * cos(hatchAngl + rad));$$

$$y\_strix\_end = y\_strix\_beg + (hatch * sin(hatchAngl + rad));$$

$$line\ (x\_strix\_beg, y\_strix\_beg, x\_strix\_end, y\_strix\_end);$$

$$\}$$

$$\}$$

Метод drawPodstavka(){} – відображає стійку опору. Спочатку будується основна лінія з координатами початку і знаходимо координати

кінця за допомогою суми початку координат з добутком косинусу кута нахилу для координати X , та синус кута нахилу для Y.

Наступним кроком цикл For виконує побудову штрихів подібно до реалізації у класі Stoika.

Результат виконання данного класу:

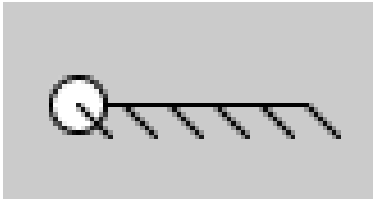


Рис.2.4

5. Огляд файлу Table

```
class Tables {
void drawTable() {
line(10, 30, 200, 30);
line(10, 80, 200, 80);
line(10, 30, 10, 240);
line(200, 30, 200, 240);
line(110, 80, 110, 240);
line(10, 160, 200, 160);
line(10, 240, 200, 240);
textSize(22);
text("Меню елементів", 15, 60);
}}
```

Створюємий метод графічно розмічає таблицю для розміщення основних елементів за допомогою вбудованих функції line().

Рис 2.5 вигляд таблиці drawTable()

6. Огляд файлу Crossing

В заданому файлі містяться методи для знаходження точок перетину ланцюгів між собою по дотичній.

$$float\ []\ peresechenie\ (float\ x1, float\ y1, float\ r1, float\ x2, float\ y2, float\ r2)\ \{$$

$$float\ d, a, h, x21, y21, xm, ym, akb;$$

$$x21 = x2 - x1;$$

$$y21 = y2 - y1;$$

$$d = sqrt(pow(x21, 2) + pow(y21, 2));$$

$$a = (pow(r1, 2) - pow(r2, 2) + pow(d, 2))/(2 * d);$$

$$h = sqrt(pow(r1, 2) - pow(a, 2));$$

$$akb = (a/(d - a));$$

$$xm = (x1 + akb * x2)/(1 + akb);$$

$$ym = (y1 + akb * y2)/(1 + akb);$$

$$P[2] = xm + h * (y2 - y1)/d;$$

$$P[3] = ym - h * (x2 - x1)/d;$$

$$P[0] = xm - h * (y2 - y1)/d;$$

$$P[1] = ym + h * (x2 - x1)/d;$$

$$return\ P;$$

$$\}$$

$$float\ []\ peresechenie\_circle\_line\ (float\ x, float\ y, float\ r, float\ xl, float\ yl,$$
$$float\ naklon)\ \{$$
$$float\ k, b, d, x1, x2, y1, y2;$$
$$k = tan(radians(naklon));$$
$$b = yl - xl * k;$$
$$d = (pow((2 * k * b - 2 * x - 2 * y * k), 2) - (4 + 4 * k * k) * (b * b - r * r + x * x + y * y - 2 * y * b));$$
$$x1 = ((-(2 * k * b - 2 * x - 2 * y * k) - sqrt(d))/(2 + 2 * k * k));$$
$$x2 = ((-(2 * k * b - 2 * x - 2 * y * k) + sqrt(d))/(2 + 2 * k * k));$$
$$y1 = k * x1 + b;$$
$$y2 = k * x2 + b;$$
$$if\ (x1 == x2)$$
$$\{$$
$$Pl[0] = x1;$$
$$Pl[1] = y1;$$
$$return\ Pl;$$
$$\}\ else$$
$$\{$$
$$Pl[0] = x1;$$
$$Pl[1] = y1;$$
$$Pl[2] = x2;$$
$$Pl[3] = y2;$$
$$return\ Pl;$$
$$\}$$
$$\}$$

$$float\ []\ petesechenie\_line\_line\ (float\ k1, float\ b1, float\ \ k2, float\ \ b2)\ \{$$
$$Per\_lines[0] = (b2 - b1)/(k1 - k2);$$

$$Per\_lines[1] = k1 * ((b2 - b1)/(k1 - k2)) + b1;$$

$$return\ Per\_lines;$$

}

7. Огляд файлу Variables

Даний файл слугує свого плану вмістилищем основних даних та змінних, які потрібні будуть для використання в подальшому

$$float\ xstart\_1 = 60;$$

$$float\ ystart\_1 = 100;$$

$$float\ xstart\_2 = 85;$$

$$float\ ystart\_2 = 170;$$

$$float\ xstart\_4 = 155;$$

$$float\ ystart\_4 = 200;$$

$$float\ Xstart1 = 156;$$

$$float\ Ystart1 = 120;$$

$$float\ dovzh\_1 = 50;$$

$$float\ rotation\_angle\_1 = 0;$$

$$float\ rotation\_angle\_2 = 90;$$

$$float\ radUgolPovorota\ =\ radians(rotation\_angle\_1);$$

$$float\ naklonRebra\_1 = 60;$$

$$float\ radius\_sharnira\_1 = 12;$$

$$float\ hatchAngl = radians(45);$$

$$float\ diag, oborot, sector, first\_pos, second\_pos = 0, razmax = 0;$$

$$float\ bx;$$

$$float\ by;$$

$$int\ boxSize\ =\ 75;$$

$$boolean\ overBox\ =\ false;$$

$$boolean\ locked\ =\ false;$$

$$float\ xOffset\ =\ 0.0;$$

$$float\ yOffset\ =\ 0.0;$$

$$float\ XA, YA, XB, YB, XC, YC;$$

$$float\ Fi\_stat, xl1\_end\_st, yl1\_end\_st, XB\_st, YB\_st, XC\_st, YC\_st, Mashtab\_st,$$

$$x1\_st, y1\_st, x2\_st, y2\_st, x3\_st, y3\_st, l1\_st, l2\_st, l3\_st, l4\_st;$$

$$float\ V\_A, V\_AX, V\_AY, V\_OA\_angl;$$

$$float\ V\_BA\_angl, k\_BA, b\_BA, V\_BO\_angl, k\_BO, b\_BO, V\_BX, V\_BY;$$

$$float\ V\_CB\_angl, k\_CB, b\_CB, k\_CO, b\_CO, V\_CX, V\_CY;$$

$$float\ a\_AO\_n, a\_A, a\_AO\_n\_angl, a\_AX, a\_AY;$$

$$float\ a\_BA\_n\_angl, a\_BA\_n, a\_BA\_n\_x, a\_BA\_n\_y, a\_BA\_tan\_angl,$$

$$k\_a\_BA\_tan, b\_a\_BA\_tan, a\_BO\_n\_angl, a\_BO\_n, a\_BO\_n\_x, a\_BO\_n\_y,$$

$$a\_BO\_tan\_angl, k\_a\_BO\_tan, b\_a\_BO\_tan, a\_BX, a\_BY;$$

$$float\ a\_CB\_n\_angl, a\_CB\_n, a\_CB\_n\_x, a\_CB\_n\_y, a\_CB\_tan\_angl,$$

$$k\_a\_CB\_tan, b\_a\_CB\_tan, k\_a\_CO\_tan, b\_a\_CO\_tan, a\_CX, a\_CY;$$

$$float\ pi = radians(180), StrAn = radians(45), strix = 6, lenght = 20, Radius = 7,$$

$$l1 = 40, l2 = 240, l3 = 270, l4 = 330, lenght3 = 4 * l1;$$

$$float\ a = -240, b = 60, c = -300, d = -220;$$

$$float\ x1 = 550, y1 = 350, naklon1 = 0;$$

$$float\ x2 = x1 + a, y2 = y1 + b, naklon2 = 0;$$

$$float\ x3 = x2 + c, y3 = y2 + d, naklon3 = 0;$$

$$float\ shir\_polz = 40;$$

$$float\ vis\_polz = 20;$$

$$float\ t = 0;$$

$$float\ n = 120;$$

$$float\ w = n * pi/30;$$

$$float\ step = w * 0.0166/5;$$

$$float\ plan = 1;$$

$$float\ dop\_vectora\_trigger = 1;$$

$$float\ taimer\_nazhatiya = 0;$$

$$float\ buffer\_pause\_step = 0;$$

$$float\ polozhennya = 1;$$

$$float\ framerate = 60;$$

$$int\ timer = 0;$$

$$float\ step2 = 1;$$

$$float\ Mashtab\_Vectora = 1.3;$$

$$float\ Mashtab\_Skorostey = 0.5;$$

$$float\ Mashtab\_Priskorennya = 0.04;$$

$$float\ []\ Pl\ =\ new\ float\ [4];$$

$$float\ Angl = 0;$$

$$float\ []\ tochki\ =\ new\ float\ [8];$$

$$float\ []\ Per\_lines\ =\ new\ float\ [2];$$

$$float\ []\ P\ =\ new\ float\ [4];$$

$$float\ []\ Plan\ =\ new\ float\ [2];$$

$$float\ []\ Polus\_Skorostey =\ new\ float\ [2]\ ;$$

$$float\ []\ Polus\_Priskorennya =\ new\ float\ [2]\ ;$$

$$float\ []\ ends\ =\ new\ float\ [4];$$

$$Tables\ table\ =\ new\ Tables();$$

$$Podstavka\ mainPodstavka\ =\ new\ Podstavka();$$

$$Podstavka\ podstavka2\ =\ new\ Podstavka();$$

$$Stoika\ mainStoika\ =\ new\ Stoika();$$

$$Stoika\ stoika2\ =\ new\ Stoika();$$

$$Stoika\ stoika3\ =\ new\ Stoika();$$

$$Block\ mainBlock\ =\ new\ Block();$$

$$Block\ block2\ =\ new\ Block();$$

$$Point\ pointMain\ =\ new\ Point();$$

$$Point\ point2\ =\ new\ Point();$$

$$Point\ point3\ =\ new\ Point();$$

$$boolean\ point2set\ =\ false;$$

$$boolean\ stoika2set\ =\ false;$$

$$float\ distAO, distAB, distB01, distBC, distOO1;$$

8. Огляд файлу Vector

Данний файл містить в собі функції для знаходження векторів швидкостей і прискорень точок та кута їх напрямку.

$$void\ vector(float\ x1, float\ y1, float\ x2, float\ y2)\ \{$$

$$line(x1, y1, x2, y2);$$

$$pushMatrix();$$

$$translate(x2, y2);$$

$$float\ a\ =\ atan2(x1 - x2, y2 - y1);$$

$$rotate(a);$$

$$line(0, 0, -7, -7);$$

$$line(0, 0, 7, -7);$$

$$popMatrix();$$

$$\}$$

$$float\ \ angle\_of\_vector\ (float\ xo, float\ y0, float\ vect\_end\_x, float\ vect\_end\_y)\ \{$$

$$float\ pi = radians(180);$$

$$if\ ((vect\_end\_x - xo! = 0)\ \&\&(vect\_end\_y - y0! = 0))\ \{$$

$$Angl = atan((vect\_end\_y - y0)/(vect\_end\_x - xo));$$

$$\} else \{$$

$$atan((vect\_end\_y - y0 + 0.001)/(vect\_end\_x - xo + 0.001));$$

$$\}$$

$$if\ (((vect\_end\_y - y0) < 0)\ \&\&\ ((vect\_end\_x - xo) < 0))\ \{$$

$$Angl = -pi + Angl;$$

$$\}\ else\ if\ (((vect\_end\_y - y0) > 0)\ \&\&\ ((vect\_end\_x - xo) < 0))\ \{$$

$$Angl = pi + Angl;$$

$$\}$$

$$return\ Angl;$$

$$\}$$

$$void\ drawLines()\ \{$$

$$line(point2.getXstart(), point2.getYstart(),$$

$$point3.getXstart(), point3.getYstart());$$

$$line(stoika2.getXstart(), stoika2.getYstart(),$$

$$point2.getXstart(), point2.getYstart());$$

$$line(stoika3.getXstart(), stoika3.getYstart(),$$

$$point3.getXstart(), point3.getYstart());$$

$$line(point3.getXstart(), point3.getYstart(),$$

$$block2.getXstart(), block2.getYstart());$$

$$\}$$

9. Огляд файлу Diplom

Перш за все ініціалізується вбудований метод Setup() який запускається 1 раз для проведення попередніх налаштувань:

$$void\ setup()\ \{$$

$$fullScreen();$$

$$rectMode(RADIUS);$$

$$frameRate(60);$$

$$\}$$

$fullScreen()$;- функція яка відповідає розміру екрану.

$rectMode(RADIUS)$; - функція-модифікатор яка вибирає яким чином розташовувати координати прямокутника.

$frameRate(60)$; - частота оновлення екрану.

Після цього ми створюємо основний метод який працює впродовж всієї програми Draw():

$$if\ (mousePressed\ ==\ true)\ \{$$
$$if\ (mouseButton == LEFT)\ \{$$
$$cursor(HAND);$$
$$\}\ else\ \{$$
$$cursor(CROSS);$$
$$\}$$
$$\}\ else\ \{$$
$$cursor(ARROW);$$
$$\}$$

Даний цикл змінює зображення курсора при натисканні клавіші миші: Якщо натиснута ліва клавіша – курсор зміниться на “руку” , якщо права- на хрестик.

$$table.drawTable();$$
$$mainPodstavka.drawPodstavka();$$
$$podstavka2.drawPodstavka();$$
$$mainStoika.drawStoika();$$
$$stoika2.drawStoika();$$
$$stoika3.drawStoika();$$
$$mainBlock.drawBlock();$$
$$block2.drawBlock();$$
$$pointMain.drawPoint();$$
$$point2.drawPoint();$$
$$point3.drawPoint();$$

Методи для створення і відображення всіх елементів програми.

Далі задається умова при якій, якщо всі елементи знаходяться не на початковому положенні проводиться з'єднання КП , розрахунок довжин ланок та відстані між стійками.Проводиться аналіз на здатність конструкції на повний оберт кривошипу чи конструкція має можливість лише на секторний рух, далі за допомогою методів peresechenie() з файлу Crossing знаходиться перетин точок і з'єднання ланок механізму. Наступним кроком проводиться аналіз та розрахування планів швидкостей та прискорення.

$$if\ (podstavka2.getXstart()! = 40\ \&\&\ podstavka2.getYstart()! = 200\ \&\&\ stoika2.getXstart()! = 60\ \&\&\ stoika2.getYstart()! = 100\ \&\&\ stoika3.getXstart()! = 60\ \&\&\ stoika3.getYstart()! = 100\ \&\&$$

$$point2.getXstart()! = 156\ \&\&\ point2.getYstart()! = 120\ \&\&\ point3.getXstart()! = 156\ \&\&\ point3.getYstart()! = 120\ \&\&\ block2.getXstart()! = 155\ \&\&\ block2.getYstart()! = 200)\ \{$$

$$drawLines();$$

$$distAO = dist(stoika2.getXstart(), stoika2.getYstart(), point2.getXstart(), point2.getYstart());$$

$$distAB = dist(point2.getXstart(), point2.getYstart(), point3.getXstart(), point3.getYstart());$$

$$distB01 = dist(stoika3.getXstart(), stoika3.getYstart(), point3.getXstart(), point3.getYstart());$$

$$distBC = dist(point3.getXstart(), point3.getYstart(), block2.getXstart(), block2.getYstart());$$

$$distOO1 = dist(stoika2.getXstart(), stoika2.getYstart(), stoika3.getXstart(), stoika3.getYstart());$$

$$diag = sqrt(pow((stoika2.getXstart() - stoika3.getXstart()), 2) + pow((stoika2.getYstart() - stoika3.getYstart()), 2));$$

$$if\ ((diag + distAO <= distAB + distB01))\ \{$$

$$oborot = 1;$$

$$text(\text{"Механізм здійснює повний оберт"}, 900, 760);$$

$$\}\ else\ if\ ((distAO + distAB + distB01 < diag))\ \{$$

$$oborot = 0;$$

$$text(\text{"Недостатня довжина ланок"}, 900, 760);$$

$$\}\ else\ \{$$

$$text(\text{"Механізм здійснює секторний рух"}, 900, 760);$$

$$oborot = 0.5;$$

$$\}$$

$$peresechenie\ (stoika2.getXstart(), stoika2.getYstart(), distAO,$$

$$stoika3.getXstart(), stoika3.getYstart(), (distAB + distB01));$$

$$if\ (oborot == 1)\ \{$$

$$first\_pos = 0;$$

$$second\_pos = radians(360);$$

$$sector = radians(360 - 0.001) * t/12;$$

$$razmax = radians(360);$$

$$\}\ else\ if\ (oborot == 0.5)\ \{$$

$$float\ []\ kr\_polozh = new\ float\ [4];$$

$$kr\_polozh[0] = P[0];$$

$$kr\_polozh[1] = P[1];$$

$$kr\_polozh[2] = P[2];$$

$$kr\_polozh[3] = P[3];$$

$$angle\_of\_vector(stoika2.getXstart(), stoika2.getYstart(), kr\_polozh[0],$$

$$kr\_polozh[1]);$$

$$first\_pos = Angl + radians(360);$$

$$angle\_of\_vector(stoika2.getXstart(), stoika2.getYstart(), kr\_polozh[2],$$

$$kr\_polozh[3]);$$

$$second\_pos = Angl;$$

$$razmax = abs(first\_pos - second\_pos);$$

$$sector = second\_pos + ((razmax - 0.001) * t/12);$$

$$circle\ (kr\_polozh[0], kr\_polozh[1], Radius);$$

$$circle\ (kr\_polozh[2], kr\_polozh[3], Radius);$$

$$\}$$

$$if\ (polozhennya == 2)\ \{$$

$$step = (razmax - 0.001)/12;$$

$$sector = second\_pos + 0.00001 + timer/60 * step;$$

$$\}$$

$$if\ (oborot == 0.5)\ \{$$

$$if\ (sector <= second\_pos - 0.001)\ \{$$

$$step *= -1;$$

$$step2 *= -1;$$

$$sector = second\_pos;$$

$$timer = 0;$$

$$w *= -1;$$

$$\}\ else\ if\ (sector >= first\_pos + 0.001)\ \{$$

$$step *= -1;$$

$$step2 *= -1;$$

$$sector = first\_pos;$$

$$w *= -1;$$

$$\}$$

$$\}$$

$$XA = stoika2.getXstart() + distAO * cos(sector);$$

$$YA = stoika2.getYstart() + distAO * sin(sector);$$

$$peresechenie(XA, YA, distAB, stoika3.getXstart(), stoika3.getYstart(), distB01);$$

$$XB = P[0];$$

$$YB = P[1];$$

$$peresechenie\_circle\_line\ (P[0], P[1], distBC, point3.getXstart(), point3.getYstart(), naklon3);$$

$$XC = Pl[0];$$

$$YC = Pl[1];$$

$$Plan[0] = 1000;$$

$$Plan[1] = 400;$$

$$Polus\_Skorostey[0] = Plan[0];$$

$$Polus\_Skorostey[1] = Plan[1];$$

$$V\_A = Mashtab\_Skorostey * w * l1;$$

$$angle_{of_{vector(stoika2.getXstart(\ \ ),stoika2.getYstart(\ \ ),XA,YA)}};$$

$$V\_OA\_angl = Angl + radians(90);$$

$$V\_AX = Polus\_Skorostey[0] + V\_A * cos(V\_OA\_angl);$$

$$V\_AY = Polus\_Skorostey[1] + V\_A * sin(V\_OA\_angl);$$

$$angle\_of\_vector\ (XA, YA, XB, YB);$$

$$V_{BA_{angl}} = Angl + radians(90);$$

$$k\_BA = tan(V\_BA\_angl);$$

$$b\_BA = V\_AY - k\_BA * V\_AX\ ;$$

$$angle_{of_{vector}(stoika3.getXstart(\ \ ),stoika3.getYstart(\ \ ),XB,YB)};$$

$$V\_BO\_angl = Angl + radians(90);$$

$$k\_BO = tan(V\_BO\_angl);$$

$$b\_BO = Polus\_Skorostey[1] - k\_BO * Polus\_Skorostey[0];$$

$$petesechenie\_line\_line(k\_BA, b\_BA, k\_BO, b\_BO);$$

$$V_{BX} = Per_{lines[0]};$$

$$V\_BY = Per\_lines[1];$$

$$angle\_of\_vector\ (XB, YB, XC, YC);$$

$$V\_CB\_angl = Angl + radians(90);$$

$$k\_CB = tan(V\_CB\_angl);$$

$$b\_CB = V\_BY - k\_CB * V\_BX;$$

$$k\_CO = tan(radians(naklon3));$$

$$b\_CO = Polus\_Skorostey[1] - k\_CO * Polus\_Skorostey[0];$$

$$petesechenie\_line\_line(k\_CB, b\_CB, k\_CO, b\_CO);$$

$$V_{CX} = Per_{lines[0]};$$

$$V\_CY = Per\_lines[1];$$

$$Polus\_Priskorennya[0] = Plan[0];$$

$$Polus\_Priskorennya[1] = Plan[1];$$

$$a\_AO\_n = pow(w, 2) * distAO * Mashtab\_Priskorennya;$$

$$a\_A = a\_AO\_n;$$

$$angle_{of_{vector(stoika2.getXstart(),stoika2.getYstart(),XA,YA)}};$$

$$a\_AO\_n\_angl = Angl;$$

$$a\_AX = Polus\_Priskorennya[0] - a\_A * cos(a\_AO\_n\_angl);$$

$$a\_AY = Polus\_Priskorennya[1] - a\_A * sin(a\_AO\_n\_angl);$$

$$angle_{of_{vector}(XA,YA,XB,YB)};$$

$$a\_BA\_n\_angl = Angl;$$

$$a\_BA\_n = Mashtab\_Priskorennya * (pow((V\_BX - V\_AX), 2) + pow((V\_BY - V\_AY), 2))/distAB;$$

$$a\_BA\_n\_x = a\_AX - a\_BA\_n * cos(a\_BA\_n\_angl);$$

$$a\_BA\_n\_y = a\_AY - a\_BA\_n * sin(a\_BA\_n\_angl);$$

$$a\_BA\_tan\_angl = a\_BA\_n\_angl + radians(90);$$

$$k\_a\_BA\_tan = tan(a\_BA\_tan\_angl);$$

$$b\_a\_BA\_tan = a\_BA\_n\_y - k\_a\_BA\_tan * a\_BA\_n\_x;$$

$$angle\_of\_vector\ (stoika3.getXstart(), stoika3.getYstart(), XB, YB);$$

$$a\_BO\_n\_angl = Angl;$$

$$a\_BO\_n = Mashtab\_Priskorennya * (pow((V\_BX - Polus\_Priskorennya[0]), 2) + pow((V\_BY - Polus\_Priskorennya[1]), 2))/distB01;$$

$$a\_BO\_n\_x = Polus\_Priskorennya[0] - a\_BO\_n * cos(a\_BO\_n\_angl);$$

$$a\_BO\_n\_y = Polus\_Priskorennya[1] - a\_BO\_n * sin(a\_BO\_n\_angl);$$

$$a\_BO\_tan\_angl = a\_BO\_n\_angl + radians(90);$$

$$k\_a\_BO\_tan = tan(a\_BO\_tan\_angl);$$

$$b\_a\_BO\_tan = a\_BO\_n\_y - k\_a\_BO\_tan * a\_BO\_n\_x;$$

$$petesechenie\_line\_line(k\_a\_BA\_tan, b\_a\_BA\_tan, k\_a\_BO\_tan, b\_a\_BO\_tan);$$

$$a\_BX = Per\_lines[0];$$

$$a\_BY = Per\_lines[1];$$

$$angle\_of\_vector\ (XC, YC, XB, YB);$$

$$a\_CB\_n\_angl = Angl;$$

$$a\_CB\_n = Mashtab\_Priskorennya * (pow((V\_CX - V\_BX), 2) + pow((V\_CY - V\_BY), 2))/distBC;$$

$$a\_CB\_n\_x = a\_BX - a\_CB\_n * cos(a\_CB\_n\_angl);$$

$$a\_CB\_n\_y = a\_BY - a\_CB\_n * sin(a\_CB\_n\_angl);$$

$$a\_CB\_tan\_angl = a\_CB\_n\_angl + radians(90);$$

$$k\_a\_CB\_tan = tan(a\_CB\_tan\_angl);$$

$$b\_a\_CB\_tan = a\_CB\_n\_y - k\_a\_CB\_tan * a\_CB\_n\_x;$$

$$b\_a\_CO\_tan = Polus\_Priskorennya[1] - k\_a\_CO\_tan * Polus\_Priskorennya[0];$$

$$petesechenie\_line\_line(k\_a\_CB\_tan, b\_a\_CB\_tan, k\_a\_CO\_tan, b\_a\_CO\_tan);$$

$$a\_CX = Per\_lines[0];$$

$$a\_CY = Per\_lines[1];$$

$$if\ (plan == 1)\ \{$$

$$vector(Polus\_Skorostey[0], Polus\_Skorostey[1], V\_CX, V\_CY);$$

$$vector(Polus\_Skorostey[0], Polus\_Skorostey[1], V\_BX, V\_BY);$$

$$vector(Polus\_Skorostey[0], Polus\_Skorostey[1], V\_AX, V\_AY);$$

$$text("Pv", Polus\_Skorostey[0], Polus\_Skorostey[1] - 7);$$

$$text("A", V\_AX, V\_AY - 5);$$

$$text("B", V\_BX, V\_BY - 5);$$

$$text("C", V\_CX, V\_CY - 5);$$

$$text("V\ A = " + V\_A/Mashtab\_Skorostey + "мм/с", 50, 480);$$

$$text("V\ B = " + sqrt(pow(V\_BX - Plan[0], 2) + pow(V\_BY - Plan[1], 2))/Mashtab\_Skorostey + "мм/с", 50, 520);$$

$$text("V\ C$$
$$= " + sqrt(pow(V\_CX - Plan[0], 2) + pow(V\_CX$$
$$- Plan[1], 2))/Mashtab\_Skorostey + "мм/с", 50, 560);$$

$$\}\}$$

Після цього створюються функціональні кнопки для:

$$rect(90, 280, 70, 20, 7);$$

$$textSize(14);$$

$$fill(0);$$

$$text("Вихід з програми", 30, 285);$$

$$fill(255);$$

$$if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX < 125\ \&\&\ mouseY$$
$$> 270\ \&\&\ mouseY < 290\ )\ \{$$

$$exit();$$

$$\}$$

Виходу з програми;

$$rect(90, 335, 70, 20, 7);$$

$$fill(0);$$

$$text("Пауза", 70, 340);$$

$$fill(255);$$

$$if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX < 125\ \&\&\ mouseY$$
$$> 325\ \&\&\ mouseY < 345\ \&\&\ ((step2! = 0)))\ \{$$

$$buffer\_pause\_step = step;$$

$$step = 0;$$

$$step2 = 0;$$

$$taimer\_nazhatiya = 20;$$

$$\}\ else\ if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX$$
$$< 125\ \&\&\ mouseY > 325\ \&\&\ mouseY < 345\ \&\&\ ((step2 =$$
$$= 0)))\ \{$$

$$step = buffer\_pause\_step;$$

$$step2 = 1;$$

$$taimer\_nazhatiya = 20;$$

$$\}$$

Зупинення руху механізму:

$$rect(90, 335, 70, 20, 7);$$

$$fill(0);$$

$$text("Пауза", 70, 340);$$

$$fill(255);$$

$$if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX < 125\ \&\&\ mouseY > 325\ \&\&\ mouseY < 345\ \&\&\ ((step2! = 0)))\ \{$$

$$buffer\_pause\_step = step;$$

$$step = 0;$$

$$step2 = 0;$$

$$taimer\_nazhatiya = 20;$$

$$\}\ else\ if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX < 125\ \&\&\ mouseY > 325\ \&\&\ mouseY < 345\ \&\&\ ((step2 = = 0)))\ \{$$

$$step = buffer\_pause\_step;$$

$$step2 = 1;$$

$$taimer\_nazhatiya = 20;$$

$$\}$$

Встановлення нульового положення механізму:

$$rect(90, 390, 70, 20, 7);$$

$$fill(0);$$

$$text("Нуль", 75, 395);$$

$$fill(255);$$

$$if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX < 125\ \&\&\ mouseY > 380\ \&\&\ mouseY < 400\ \&\&\ (taimer\_nazhatiya < 0))\ \{$$

$$t = 0;$$

$$timer = 0;$$

$$\}$$

Зміна відображення аналізу між анімацією, статичним зображенням та покроковим:

$$rect(90, 455, 70, 20, 7);$$

$$fill(0);$$

$$text("Анім./стат./крок", 35, 460);$$

$$fill(255);$$

$$if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX < 125\ \&\&\ mouseY > 445\ \&\&\ mouseY < 465\ \&\&(taimer\_nazhatiya < 0)\ \&\&\ (polozhennya == 3))$$

$$\{$$

$$polozhennya = 2;$$

$$taimer\_nazhatiya = 20;$$

$$t = 0;$$

$$timer = 0;$$

$$\}\ else\ if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX < 125\ \&\&\ mouseY > 445\ \&\&\ mouseY < 465\ \&\&\ (taimer\_nazhatiya < 0)\ \&\&\ (polozhennya == 2))$$

$$\{$$

$$polozhennya = 1;$$

$$taimer\_nazhatiya = 20;$$

$$t = 0;$$

$$timer = 0;$$

$$step = w * 0.0166/5;$$

$$\}\ else\ if\ ((mousePressed)\ \&\&\ mouseX > 55\ \&\&\ mouseX < 125\ \&\&\ mouseY > 445\ \&\&\ mouseY < 465\ \&\&\ (taimer\_nazhatiya < 0)\ \&\&\ (polozhennya == 1))$$

$$\{$$

$$polozhennya = 3;$$

$$taimer\_nazhatiya = 20;$$

$$\}$$

Також для кожного створюємого елемента відбувається перевірка на те чи знаходить курсор в радіусі положення об’єкту, у позитивному випадку колір механізму змінюється :

$$if\ (mouseX > pointMain.getXstart() - pointMain.getRadius()\ \&\&\ mouseX < pointMain.getXstart() + pointMain.getRadius()\ \&\&$$

$$mouseY > pointMain.getYstart() - pointMain.getRadius()\ \&\&\ mouseY < pointMain.getYstart() + pointMain.getRadius())\ \{$$

$$overBox = true;$$

$$if\ (!locked)\ \{$$

$$stroke(0);$$

$$fill(153);$$

$$\}\ldots$$

$$\ldots else\ \{$$

$$stroke(0);$$

$$fill(255);$$

$$overBox = false;$$

$$\}$$

Вбудована функція mousePressed() за допомогою якої:

При натисканні на елементи в меню елементів, створюються копії елементів для роботи з ними в подальшому, а також для вибору об’єкта і переміщення його по робочому екрану.

$$if\ (overBox)\ \{$$

$$locked = true;$$

$$fill(255, 255, 255);$$

$$\}\ else\ \{$$

```
                    locked = false;
                }
if (mouseX
        > mainStoika.getXstart()
        - mainStoika.getRadius() && mouseX
        < mainStoika.getXstart() + mainStoika.getRadius() &&
  mouseY
        > mainStoika.getYstart()
        - mainStoika.getRadius() && mouseY
        < mainStoika.getYstart() + mainStoika.getRadius()) {
                    if (stoika2set == false) {
                    stoika2.setXstart(width/2.0);
                    stoika2.setYstart(height/2.0);
                    stoika2set = true;
                    delay(500);
                    } else {
                    stoika3.setXstart(width/2.0);
                    stoika3.setYstart(height/2.0);
                    delay(500);
                    }
        xOffset = mouseX - mainStoika.getXstart();
        yOffset = mouseY - mainStoika.getYstart();
 } else if (mouseX
        > pointMain.getXstart()
        - pointMain.getRadius() && mouseX
        < pointMain.getXstart() + pointMain.getRadius() &&
   mouseY
        > pointMain.getYstart()
        - pointMain.getRadius() && mouseY
        < pointMain.getYstart() + pointMain.getRadius()) {
                    if (point2set == false) {
                    point2.setXstart(width/2.0);
```

$$point2.setYstart(height/2.0);$$

$$point2set = true;$$

$$delay(500);$$

$$\} else \{$$

$$point3.setXstart(width/2.0);$$

$$point3.setYstart(height/2.0);$$

$$delay(500);$$

$$\}$$

$$xOffset = mouseX - pointMain.getXstart();$$

$$yOffset = mouseY - pointMain.getYstart();$$

$$\} else\ if\ (mouseX > point2.getXstart() - point2.getRadius()\ \&\&\ mouseX < point2.getXstart() + point2.getRadius()\ \&\&$$

$$mouseY > point2.getYstart() - point2.getRadius()\ \&\&\ mouseY < point2.getYstart() + point2.getRadius())\ \{$$

$$xOffset = mouseX - point2.getXstart();$$

$$yOffset = mouseY - point2.getYstart();$$

$$\} else\ if\ (mouseX > point3.getXstart() - point3.getRadius()\ \&\&\ mouseX < point3.getXstart() + point3.getRadius()\ \&\&$$

$$mouseY > point3.getYstart() - point3.getRadius()\ \&\&\ mouseY < point3.getYstart() + point3.getRadius())\ \{$$

$$xOffset = mouseX - point3.getXstart();$$

$$yOffset = mouseY - point3.getYstart();$$

$$\} else\ if\ (mouseX > mainBlock.getXstart() - mainBlock.getRadius()\ \&\&\ mouseX < mainBlock.getXstart() + mainBlock.getRadius()\ \&\&$$

$$mouseY > mainBlock.getYstart() - mainBlock.getRadius()\ \&\&\ mouseY < mainBlock.getYstart() + mainBlock.getRadius())\ \{$$

$$block2.setXstart(width/2.0);$$

$$block2.setYstart(height/2.0);$$

$$xOffset = mouseX - mainBlock.getXstart();$$

$$yOffset = mouseY - mainBlock.getYstart();$$

$$delay(500);$$

$$\} else\ if\ (mouseX > block2.getXstart() - block2.getRadius()\ \&\&\ mouseX < block2.getXstart() + block2.getRadius()\ \&\&$$

$$mouseY > block2.getYstart() - block2.getRadius()\ \&\&\ mouseY < block2.getYstart() + block2.getRadius()) \{$$

$$xOffset = mouseX - block2.getXstart();$$

$$yOffset = mouseY - block2.getYstart();$$

$$\} else\ if\ (mouseX > mainPodstavka.getXstart() - mainPodstavka.getRadius()\ \&\&\ mouseX < mainPodstavka.getXstart() + mainPodstavka.getRadius()\ \&\&$$

$$mouseY > mainPodstavka.getYstart() - mainPodstavka.getRadius()\ \&\&\ mouseY < mainPodstavka.getYstart() + mainPodstavka.getRadius()) \{$$

$$podstavka2.setXstart(width/2.0);$$

$$podstavka2.setYstart(height/2.0);$$

$$podstavka2.setLengt(200.0);$$

$$xOffset = mouseX - podstavka2.getXstart();$$

$$yOffset = mouseY - podstavka2.getYstart();$$

$$\} else\ if\ (mouseX > podstavka2.getXstart() - podstavka2.getRadius()\ \&\&\ mouseX < podstavka2.getXstart() + podstavka2.getRadius()\ \&\&$$

$$mouseY > podstavka2.getYstart() - podstavka2.getRadius() \&\& mouseY < podstavka2.getYstart() + podstavka2.getRadius()) \{$$

$$xOffset = mouseX - podstavka2.getXstart();$$

$$yOffset = mouseY - podstavka2.getYstart();$$

$$\}$$

$$\} else\ if\ (mouseButton == RIGHT) \{$$

$$\}$$

$$\}$$

Вбудована функція mouseDragged() яка реагує при натисканні і утриманні кнопки миші, з її допомогою виконується переміщення об'єкту при умові що курсор знаходиться в межах радіусу точки для захоплення,

$$void\ mouseDragged() \{$$

$$if\ (locked) \{$$

$$if\ (mouseX > stoika2.getXstart() - stoika2.getRadius() \&\& mouseX < stoika2.getXstart() + stoika2.getRadius() \&\&$$

$$mouseY > stoika2.getYstart() - stoika2.getRadius() \&\& mouseY < stoika2.getYstart() + stoika2.getRadius()) \{$$

$$stoika2.setXstart(mouseX - xOffset);$$

$$stoika2.setYstart(mouseY - yOffset);$$

$$\} else\ if\ (mouseX > stoika3.getXstart() - stoika3.getRadius() \&\& mouseX < stoika3.getXstart() + stoika3.getRadius() \&\&$$

$$mouseY > stoika3.getYstart() - stoika3.getRadius() \&\& mouseY < stoika3.getYstart() + stoika3.getRadius()) \{$$

$$stoika3.setXstart(mouseX - xOffset);$$

$$stoika3.setYstart(mouseY - yOffset);$$

$$\} else\ if\ (mouseX > point2.getXstart() - point2.getRadius() \&\& mouseX < point2.getXstart() + point2.getRadius() \&\&$$

$$mouseY > point2.getYstart() - point2.getRadius() \&\& mouseY < point2.getYstart() + point2.getRadius()) \{$$

$$point2.setXstart(mouseX - xOffset);$$

$$point2.setYstart(mouseY - yOffset);$$

$$\} else\ if\ (mouseX > point3.getXstart() - point3.getRadius() \&\& mouseX < point3.getXstart() + point3.getRadius() \&\&$$

$$mouseY > point3.getYstart() - point3.getRadius() \&\& mouseY < point3.getYstart() + point3.getRadius()) \{$$

$$point3.setXstart(mouseX - xOffset);$$

$$point3.setYstart(mouseY - yOffset);$$

$$\} else\ if\ (mouseX > block2.getXstart() - block2.getRadius() \&\& mouseX < block2.getXstart() + block2.getRadius() \&\&$$

$$mouseY > block2.getYstart() - block2.getRadius() \&\& mouseY < block2.getYstart() + block2.getRadius()) \{$$

$$block2.setXstart(mouseX - xOffset);$$

$$block2.setYstart(mouseY - yOffset);$$

$$\} else\ if\ (mouseX > podstavka2.getXstart() - podstavka2.getRadius() \&\& mouseX < podstavka2.getXstart() + podstavka2.getRadius() \&\&$$

$$mouseY > podstavka2.getYstart() - podstavka2.getRadius() \&\& mouseY < podstavka2.getYstart() + podstavka2.getRadius()) \{$$

$$podstavka2.setXstart(mouseX - xOffset);$$

$$podstavka2.setYstart(mouseY - yOffset);$$

$$\}\}\}$$

Вбудована функція mouseReleased() яка виконується при умові коли будь-яка кнопка миші відпускається.

$$void\ mouseReleased() \{$$

$$locked = false; \}$$

Результат виконання програми

При запуску програми ми можемо бачити меню елементів та функціональні клавіші для додаткової роботи з програмою.

Для завершення роботи з програмою є відповідна кнопка : Вихід з програми;

Для зупинення та відновлення руху механізму в будь-який момент – кнопка пауза;

Для встановлення механізму в вихідне положення – кнопка Нуль;

Для зображення плану швидкостей або прискорення на вибір – кнопка План Pv/Pa.

Рис.2.6 вигляд програми при запуску

Після створення механізму програма автоматично почне рух механізму, зобразить в правій нижній частині екрану, який рух виконує шарнір ,та автоматично побудує план швидкостей та відобразить їх на екрані.

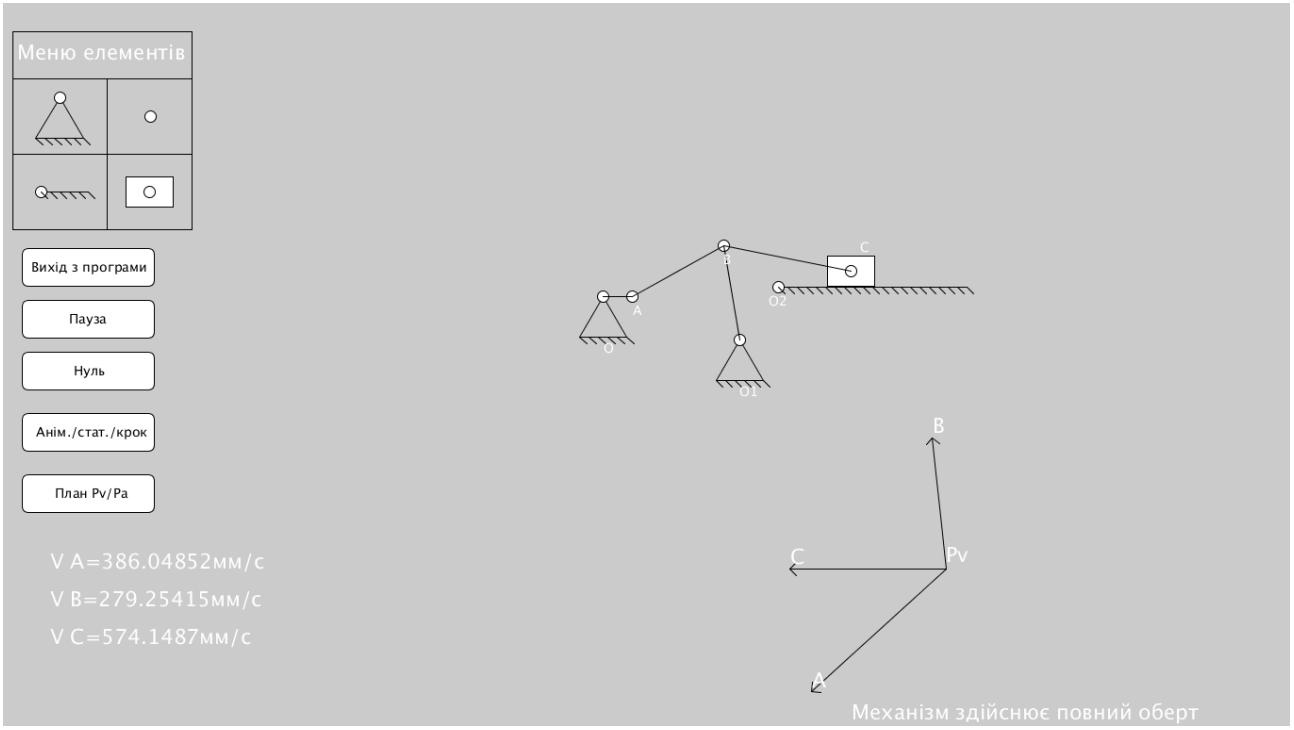


Рис.2.7

Також можна переглянути план прискорення натиснувши відповідну кнопку -План Pv/Pa.

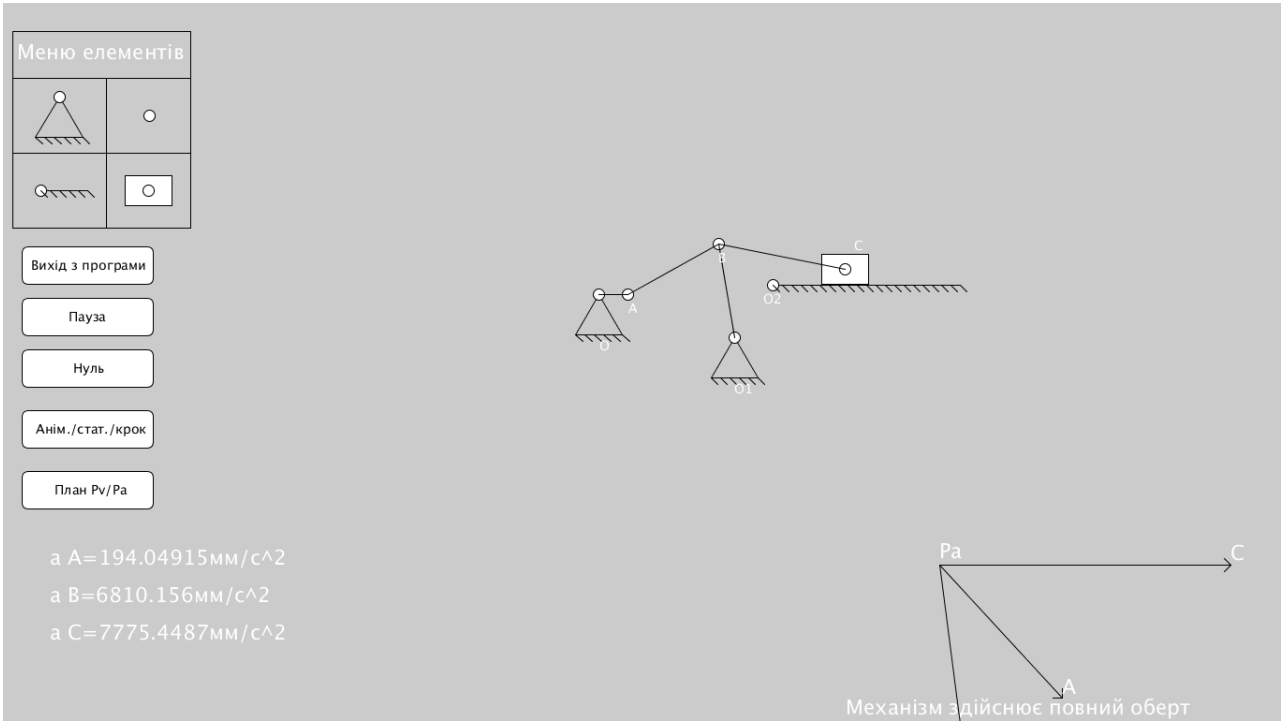


Рис2.8.

Висновок

В роботі було розглянуто основні положення та теоретичні відомості щодо побудови та принципи аналізу плоских важільних механізмів.

Наведено на конкретних прикладах способи структурного та кінематичного дослідження, знаходження числа ступеней свободи механізму, дослідження та побудова плану руху, швидкостей та прискорень.

Реалізовано та протестовано програмне забезпечення яке забезпечує автоматизацію процесу збору механізму, автоматичного його аналізу та розрахунку векторів швидкостей та прискорень.

ПОСИЛАННЯ

1. Теорія механізмів і машин. Курс лекцій для студентів спеціальності „Динаміка і міцність машин”/ Автор: к.т.н., доц. О.П. Заховайко. – К.: НТУУ "КПІ", 2010. – 243 с.

2. Теорія механізмів і машин: Інтерактивний комплекс навчально-методичного забезпечення – Рівне: НУВГП,2006. – 160 с.

3. Теорія механізмів і машин. Частина1: навчальний посібник/Укл. В.В. Пирогов, Г.Б. Філімоніхін, Ю.А. Невдаха. – Кропивницький: ЦНТУ, 2017. – 88с.

4. Ступінь рухливості кінематичного ланцюга [Електронний ресурс] – Режим доступу до ресурсу: https://stud.com.ua/72487/tehnika/stupin_ruhlivosti_kinematichnogo_lantsyuga.

5. Бондар П. М., Мироненко П. С. Дослідження механізмів. Лабораторний практикум для студентів спеціальності "Прилади і системи орієнтації та навігації", – електронне видання, 2011, 45с. ISBN