УДК 681.5

В. Смолій, Н. Смолій, О. Лісовиченко

ПРОГРАМНО-АПАРАТНИЙ КОМПЛЕКС ДЛЯ КЕРУВАННЯ МОДУЛЬНИМИ МАНІПУЛЯТОРАМИ

Анотація: Запропоновано платформу для програмування і підналагодження алгоритмів інверсної кінематики для створюваних користувачем кінематичних ланцюгів.

Ключові слова: Модульні роботи, кінематичний ланцюг, c++, opengl.

Опис проблеми

Створюваний програмно-апаратний комплекс для керування модульними маніпуляторами дає можливість застосування даного виду пристроїв в різних галузях виробництва від зборки компонентів друкованих плат до операцій зварювання та переносу, в залежності від використаних при побудові робота компонентів. Програмний комплекс надаватиме можливість незалежно від мети використання налагоджувати роботу будь якого пристрою [1].

Розглядається кінематична пара, як рухоме сполучення двох твердих ланок, що накладає обмеження на їх відносний рух умовами накладеної в'язі. Кожна з умов в'язі усуває один ступінь свободи, тобто можливість одного з 6 незалежних відносних рухів у просторі. В прямокутній системі координат можливі 3 поступальних рухи і 3 обертальних.

Сутність алгоритму прямої кінематики полягає в тому, що керуючий вплив розповсюджується від батьківських елементів до дочірніх. Для алгоритму інверсної кінематики - знаючи кінцеву орієнтацію та положення елемента кінематичного ланцюга, відбувається розрахунок відповідних параметрів його батьківських елементів.

Задача досліджень полягає в розробці програмного забезпечення керування модульними маніпуляторами.

Рішення вказаної задачі досягається тим, що розроблений програмний комплекс застосовується для моделювання та імітації роботи маніпуляторів будь-якої конструкції та призначення.

Метою роботи є створення, систематизація та вдосконалення методів моделювання маніпуляторів [2, 3].

Основний зміст і результати роботи

Реалізація запропонованого програмно-апаратного комплексу для керування модульними маніпуляторами здійснена мовою C++ з використанням бібліотеки OpenGL [4].

Код створення ланцюга спирається на створення об’єкту керуючого класу та використанні його методів для створення елементів кінематичного ланцюга. Наприклад,

 ISSN 1560-8956

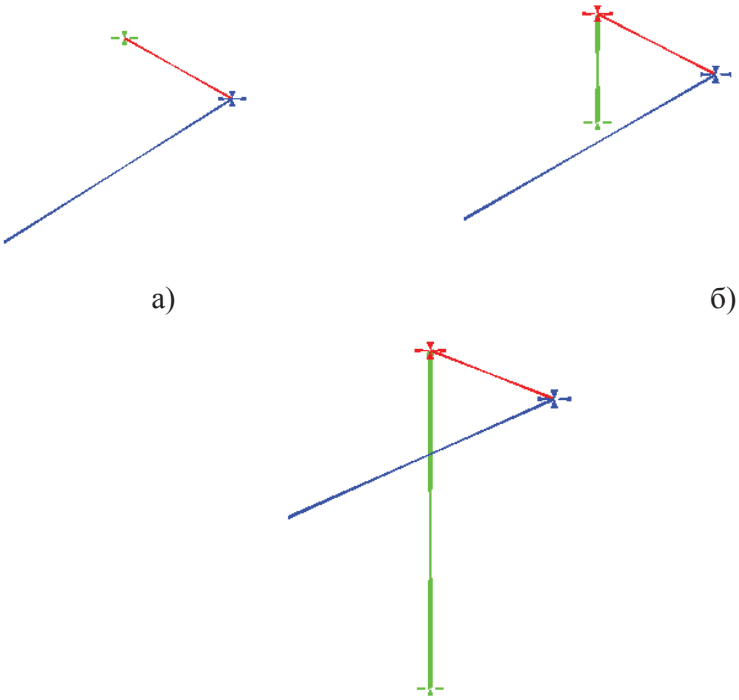
для лінійних елементів ланки, слід вказати початкову позицію, напрямок руху, максимальне видовження та початкове видовження.

```
skeleton test_skelet;
test_skelet.generate_linear_joint(glm::vec3(-1.0,-1.0,-1.0), glm::vec3(0.0f,1.0f,0.0f), 2.0f, 0.0f);
test_skelet.generate_linear_joint(glm::vec3(-1.0,-1.0,-1.0), glm::vec3(1.0f,0.0f,0.0f), 2.0f, 0.0f,0);
test_skelet.generate_linear_joint(glm::vec3(-1.0,-1.0,-1.0), glm::vec3(0.0f,0.0f,1.0f), 2.0f, 0.0f,1);
```

В наведеному уривку проілюстровано використання методу generate_linear_joint, який додає ланцюг поступального руху в кінематичний ланцюг test_skelet. Для додавання необхідно вказати початкову позицію фіксації, вектор, в якому буде відбуватися рух, максимальне видовження, початкове видовження, та індекс вузла до якого буде під’єднаний даний вузол.

Надалі керування в ручному режимі здійснюється за допомогою методів void joint::apply_transformation (float strength) відповідних класів (в нашому випадку класу linear_joint). На які в якості сили впливу подається значення за замовчуванням, що дорівнює 0.002.

На рис. 1 зображені варіанти руху (пересування) зв’язаних ланок кінематичного ланцюга. На рис. 1, а наведені деталі реалізації руху знизу вгору відповідної ланки.



а) б)

в)

Рис. 1. Приклади руху ланок кінематичного ланцюга

На рис. 1, б додано керуюче переміщення зеленою ланкою відповідних червоної та синіх ланок.

Рис. 1, в містить ілюстрацію подальшого руху об’єкту в обраному напрямку.

Повний алгоритм роботи поточної версії програми проілюстровано на рис. 2. Наведено ініціювання OpenGL, налаштування користувацького вводу-виводу та формування буферів для візуалізації складових. Після виконання основних функцій в алгоритмі передбачені заходи коректного завершення роботи зі звільненням задіяних об’ємів пам’яті.

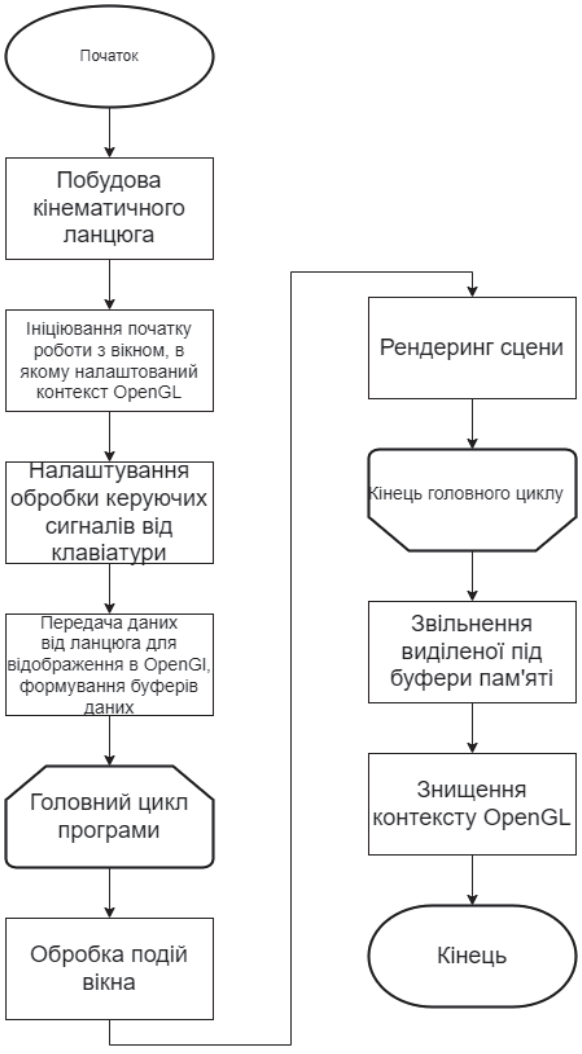


Рис. 2. Узагальнена блок-схема алгоритму роботи програми

Програмна реалізація передбачає використання декількох створених користувачем екземплярів класів, які керуються за допомогою екземпляру керуючого класу skeleton з відповідними властивостями для відбудови руху кінематичного ланцюга [4-7].

ISSN 1560-8956

Joint зберігає узагальнені атрибути та методи для класів Rotation_joint та Linear_joint.

У програмі було реалізовано відповудну схему класів.

Клас joint є батьківським класом для класів linear joint та rotational join, які в свою чергу використовуються об’єктами класу skeleton ,який відповідає за зберігання інформації про створюваний кінематичний ланцюг і використовується програмою для керування [8-11].

Прототипи даних класів:

```
class joint{
protected:
	std::vector<joint*> attached_joints;
public:
	uint8_t type;
	glm::vec3 position;
	glm::vec3 orientation;
	float length;
	void attach(joint*);
	void detach(uint32_t);
	//normal vector to the surface of rotation? angle of rotation(rads)
	void recursive_rotate(glm::mat3,float);
	//unit vector of move, distance
	void recursive_move(glm::vec3,float);
	//
	virtual void apply_transformation(float);
};
class rotation_joint: public joint{
protected:
	glm::vec2 rotation_limits;
	glm::vec3 rotation_axis;
	glm::mat3 rotation_matrix;
	float current_state;
	glm::mat3 gen_rotation_matrix();
public:
	rotation_joint(glm::vec3, glm::vec3, glm::vec2, float);
	void apply_transformation(float);
};
class linear_joint: public joint{
```

```
protected:
        glm::vec3 move_direction;
        float move_limit;
public:
        float current_ext;
        linear_joint(glm::vec3,glm::vec3,float,float);
        int virtual_test();
        void apply_transformation(float);
};
class skeleton{
private:
public:
        std::vector<joint*> joints;
        void generate_rotation_joint(glm::vec3, glm::vec3, glm::vec2,float,int32_t def_attach
= -1);
        void generate_linear_joint(glm::vec3, glm::vec3, float,float,int32_t def_attach = -1);
        void skeleton_handle_key_presses(uint8_t*);
        std::pair<std::vector<GLfloat>,std::vector<GLushort>> gather_joint_data();
        ~skeleton();
};
```

Через клас skeleton здійснюється контроль за станами доданих користувачами вузлів.

Описана програма дозволяє розробити метод керування роботами довільної конструкції у вигляді програмного забезпечення для моделювання їх конструкції.

Висновки

Запропонований програмно-апаратний комплекс, який спирається на використання описаної ієрархії класів, дозволяє моделювати та імітувати роботу маніпуляторів будь-якої конструкції та призначення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. https://www.learnaboutrobots.com/forwardKinematics.html

2. *Кіницький Я. Т.* Теорія механізмів і машин: Підручник. — К.: Наукова думка,2002. — 660 с.

3. *Кореняко О. С.* Теорія механізмів і машин: Навчальний посібник / За ред. Афанасьєва М. К. — К.: Вища школа, 1987.

4. *Попов С. В., Бучинський М. Я., Гнітько С. М., Чернявський А. М.* Теорія механізмів технологічних машин: підручник для студентів механічних спеціальностей закладів вищої освіти. Харків: — НТМТ, 2019. — 268 с.

 ISSN 1560-8956

5. *Смолий В. Н.* Системное моделирование электронных аппаратов и компонентов / *В. Н. Смолий* // Адаптивні системи автоматичного управління : міжвідомчий науково-технічний збірник. – 2015. – № 1(26). – С. 128–136. – Бібліогр.: 8 назв.

6. *Смолий В. Н.* Особенности моделирования электронных аппаратов различного назначения и условий эксплуатации / *В. Н. Смолий* // Адаптивні системи автоматичного управління : міжвідомчий науково-технічний збірник. – 2016. – № 1(28). – С. 116–128. – Бібліогр.: 12 назв.

7. *Смолий В. Н.* Комплексирование технических средств производства электронных аппаратов различного назначения и условий эксплуатации / *В. Н. Смолий* // Адаптивні системи автоматичного управління : міжвідомчий науково-технічний збірник. – 2017. – № 1(30). – С. 147–163. – Бібліогр.: 16 назв.

8. *Smoliy V.* Management conception designer preproduction of electronic vehicles / *V. Smoliy* // Адаптивні системи автоматичного управління : міжвідомчий науково-технічний збірник. – 2019. – № 1 (34). – С. 113–124. – Бібліогр.: 22 назви.

9. *Смолий В. М.* Модель автоматизованого управління конструкторською підготовкою виробництва електронних апаратів / *В.М. Смолій, Н.В. Смолій* // Адаптивні системи автоматичного управління : міжвідомчий науково-технічний збірник. – 2022. – № 1(40). – С. 129–133. – Бібліогр.: 5 назв.

10. *Smolij N.* Simulation tools: formal language for cellular automatons behavior description / *N. Smolij, O. Lisovychenko, V. Smolij* // Адаптивні системи автоматичного управління : міжвідомчий науково-технічний збірник. – 2022. – № 2(41). – С. 16–21. – Бібліогр.: 3 назв.

11. *Smolij V.* Graphical shell for constructing user-entered arithmetic functions / *V. Smolij, O. Lisovychenko, N. Smolij* // Адаптивні системи автоматичного управління : міжвідомчий науково-технічний збірник. – 2023. – № 1(42). – С. 115–120. – Бібліогр.: 5 назв.