**Выводы.**

В ходе проведенных исследований предложен классификатор типовых сварных конструкций, широко применяемых в приборостроении и машиностроении. Дан перечень стандарных универсально-сборных приспособлений для сборочно-сварочных работ, а также граф декомпозиции приспособлений на основе данных элементов с точки зрения автоматизации проектирования. Граф разбит на основные группы. На базе данного графа разработана параметрическая модель, описывающая ход выполнения технологической операции по изготовлению сварных конструкций с применением УСПС.

Предложенная параметрическая модель является основой для математической модели, которая даст возможность разработать модуль автоматизированного проектирования приспособлений для сборки и сварки сварных конструкций на основе стандартизованного многократного использования деталей и сборочных единиц УСПС.

**СПИСОК ЛИТЕРАТУРЫ:**

*1. Подобедов, В.В. Состояние, перспективы и концепция развития сборочно-сварочной оснастки в машиностроении [Текст] / В.В. Подобедов, В.И. Роменский // Резание и инструмент в технологических системах. – 2000. – №58. – С. 34-38.*

*2. Филатов, Л.С. Разработка и внедрение переналаживаемой технологической оснастки для сборочно-сварочного производства [Текст] / Л.С. Филатов // Резание и инструмент в технологических системах. – 2000. – №58. – С. 63-64.*

*3. Роменский, В.И. Исследование плотности рассеивания сварочных брызг на рабочие поверхности сборочно-сварочной оснастки при выполнении сварочных операций [Текст] / В.И. Роменский // Вестник НТУУ "ХПИ". – 2000. – №110. – С. 14-18.*

4. *Спосіб випробування й оцінка якості захисних покрить від дії зварювальних бризок [Текст]: деклараційний патент 46995 А Україна: МПК7 G 01 N 19/06, G 01 N 25/00 / Подобєдов В.В., Роменський В.І. та ін.; заявник і патентовласник Державне підприємство «Завод імені В.О. Малишева»; заявл. 03.05.01; опубл. 17.06.02, Бюл. № 6. – 3 с.*

*УДК 681.3.07*

# МЕТОДИ ПОШУКУ ОПТИМАЛЬНОГО УПРАВЛІННЯ

*А.І. Бронніков, Харківський національний університет радіоелектроніки*

*Для планування переміщення в програму об'єкта повинен бути закладений алгоритм, за яким приймається рішення щодо планування переміщення. На підставі методу динамічного програмування розроблено підхід для вирішення завдання пошуку оптимального управління, який реалізований програмно.*

*Для планирования перемещения в программу объекта должен быть заложен алгоритм, по которому принимается решение о планировании перемещения. На основании метода динамического программирования разработан подход для решения задачи поиска оптимального управления, который реализован программно.*

*In order to plan the object moving in the program should be laid the algorithm by which the decision on the planning movement is acceptedt. Based on dynamic programming approach solved the problem of finding of the optimal control, which is implemented in software.*

***Ключові слова:*** *динамічне програмування, оптимізація переміщення, оптимальне управління, методи оптимального управління*

**Введення**

Розроблена достатня кількість математичних методів і алгоритмів для вирішення завдань пошуку оптимального шляху проходження рухомого об'єкту.

Одним з перспективних методів є метод динамічного програмування.

Динамічне програмування – розділ оптимального програмування (оптимального управління), в якому процес прийняття рішення та управління, може бути розбитий на окремі етапи (кроки).

Динамічне програмування дозволяє звести одну складну задачу з багатьма змінними до багатьох задач з малим числом змінних. Це значно скорочує обсяг обчислень і прискорює процес прийняття управлінського рішення.

Управління – сукупність рішень, прийнятих на кожному етапі для впливу на хід розвитку процесу.

Операція – керований процес, тобто можна вибирати якісь параметри, що впливають на хід процесу і управляти кроками операції, забезпечувати виграші на кожному кроці і в цілому за операцію.

Рішення на кожному кроці називається «кроковим управлінням».

Сукупність усіх крокових управлінь являє собою управління операцією в цілому [1].

**Постановка задачі динамічного програмування**

Потрібно знайти таке управління, при якому вартість вироблених переміщень зверталася б до мінімум:

$$F(x) = \sum_{i=1}^{m} F_i(x_i) \longrightarrow \min \qquad (1)$$

де $F$ – вартість за операцію;

$F_i(x_i)$ – вартість і-го кроку;

$x$ – управління операцією в цілому;

$x_i$ – управління на і-му кроці (i = 1,2, ..., m).

У загальному випадку крокові управління $(x_1, x_2 \ldots x_m)$ можуть стати числами, векторами, функціями.

Управління ($x^*$), при якому досягається мінімум, називається оптимальним керуванням. Оптимальність управління складається із сукупності оптимальних крокових управлінь $x^* = x_1^*, x_2^*, \ldots, x_m^*$.

$F^* = \min\{F^*(x^*)\}$ – мінімальна вартість, яка досягається при оптимальному управлінні $x^*$.

Виходячи з умов кожного конкретного завдання, довжину кроку вибирають таким чином, щоб на кожному

кроці отримати просту задачу оптимізації та забезпечити необхідну точність обчислень.

Основним методом динамічного програмування є метод рекурентних співвідношень; який заснований на використанні принципу оптимальності, розробленого американським математиком Р. Беллманом.

Суть принципу полягає в наступному. Які б не були початковий стан на будь-якому кроці і управління, вбрання на цьому кроці, наступні управління повинні вибиратися оптимальними щодо стану, до якого прийде система в кінці кожного кроку.

Використання даного принципу гарантує, що управління, вбрання на будь-якому кроці, краще не локально, а з точки зору процесу в цілому.

Умовна оптимізація:

$$S_0 \xrightarrow[шаг]{x_1} S_1 \xrightarrow[шаг]{x_2} S_2 \xrightarrow[шаг]{x_3} S_3 \ldots \xrightarrow[шаг]{x_m} S_m \qquad (2)$$

Безумовна оптимізація:

$s_i$ – стан системи на і-му кроці. Основна рекуррентна формула динамічного програмування у разі вирішення завдання максимізації має вигляд:

$$f_m(i) = \min \begin{Bmatrix} f_m(вартість\ \ шагу) + \\ + f_{m+1}(новий\ стан\ перед\ шагом\ m+1) \end{Bmatrix}, \qquad (3)$$

де мінімум в даній формулі береться по всіх можливих рішень в ситуації, коли система на кроці *m* знаходиться в стані *i*.

Величина $f_m(i)$ – мінімальна вартість завершення завдання переміщення зі стану *i*, якщо припустити, що на кроці *m*, система знаходиться в стані *i*.

Максимальна ефективність управління може бути отримана мінімізацією суми вартостей самого кроку *m* і мінімальної вартості кроку $(m+1)$ і далі до завершення шляху.

Плануючи багатокрокову операцію треба вибирати управління на кожному кроці з урахуванням всіх його майбутніх наслідків на ще майбутніх кроках.

Управління на *i*-му кроці вибирається не так, щоб вартість саме даного кроку була мінімальна, а так, щоб була мінімальна сума вартостей на всіх кроках, що залишилися, плюс даний крок.

Серед усіх кроків останній крок планується без оглядки на майбутнє, тобто щоб він сам, приніс найбільшу вигоду і найменшу вартість.

Завдання динамічного програмування вирішується з кінця, тобто з останнього кроку. Вирішується завдання в 2 етапи:

1 етап (від кінця до початку по кроках).

Проводиться умовна оптимізація, в результаті чого знаходяться умовні оптимальні управління і умовні вартості по всіх кроках процесу.

2 етап (від початку до кінця по кроках).

Вибираються (прочитуються) вже готові рекомендації від 1-го кроку до останнього і знаходиться безумовне оптимальне керування $x^*$, $x^* = x_1^*, x_2^*, \ldots, x_m^*$.

При постановці завдань динамічного програмування слід керуватися такими принципами:

а) вибрати параметри (фазові координати), що характеризують стан *S* керованої системи перед кожним кроком;

б) розчленувати операцію на етапи (кроки);

в) з'ясувати набір крокових управлінь $x_i$ для кожного кроку і накласти на них обмеження;

г) визначити яку вартість має і-ий крок управління $x_i$, якщо перед цим система була в стані *S*, тобто записати «вартісну функцію»:

$$W_i = f_i(S, x_i) \qquad (4)$$

д) визначити, як змінюється стан *S* системи *S* під впливом управління $x_i$ на і-му кроці: воно переходить в новий стан

$$S' = \varphi_i(S, x_i) \qquad (5)$$

е) записати основне рекурентне співвідношення динамічного програмування, що виражає умовну оптимальну вартість $W_i(s)$ (починаючи з і-го кроку і до кінця) через вже відому функцію $W_{i+1}(s)$:

$$W_i(S) = \min_{x_i} \{ f_i(S, x_i) + W_{i+1}(\varphi_i(S, x_i)) \} \qquad (6)$$

Цій вартості відповідає умовне оптимальне керування на і-му кроці $x_i(s)$ (причому у вже відому функцію $W_{i+1}(s)$ треба замість *S* підставити змінений стан $S' = \varphi_i(S, x_i)$;

ж) провести умовну оптимізацію останнього (m-го) кроку, задаючись гамою станів *S*, з яких можна за один крок дійти до кінцевого стану, обчислюючи для кожного з них умовний оптимальний виграш за формулою $W_m(S) = \min_{x_m} \{ f_m(S, x_m) \}$;

з) провести умовну оптимізацію (m-1)-го, (m-2)-го і т.д. кроків за формулою (6), вважаючи в ній і = (m-1), (m-2), ..., і для кожного з кроків вказати умовне оптимальне керування $x_i(s)$, при якому досягається мінімум.

і) провести безумовну оптимізацію управління, «читаючи» відповідні рекомендації на кожному кроці. Взяти знайдене оптимальне управління на першому кроці; змінити стан системи за формулою (5); для знову знайденого стану знайти оптимальне управління на другому кроці $x_2^*$ і т.д. до кінця.

**Автоматизація пошуку оптимального управління**

Для розробки програмного забезпечення алгоритмів обрано середовище для математичних розрахунків MatLab.

Розроблено виконуваний m-файл GridWorld.m для реалізації алгоритму на базі динамічного програмування.

У кожній програмі присутні наступні складові:

- визначення змінних;
- введення початкових значень і обмежень;
- реалізація алгоритму;
- візуалізація результатів.

Вхідна інформація, яка потрібна для роботи програми:

- розмір поля (змінна board);
- початкові координати об'єкта (змінна start);
- координати мети переміщення (змінна goal);

- інформація про перешкоди (змінна P).

Розглянемо покрокове виконання розробленого алгоритму на основі методу динамічного програмування. Перший крок відображений на рис.1 і відображає лише перешкоду і поле переміщення. У даному алгоритмі не важно початкова позиція об'єкта - алгоритм розраховує оптимальні траєкторії для кожної точки поля, починаючи з координат цілі (рис.2). Інтерес представляє процес пошуку виходу з обмеженої області, в якій знаходиться мета.

На рис. 3 зображений результат 5-ої ітерацій розрахунку, бачимо, що алгоритм вже поза зоною перепони.

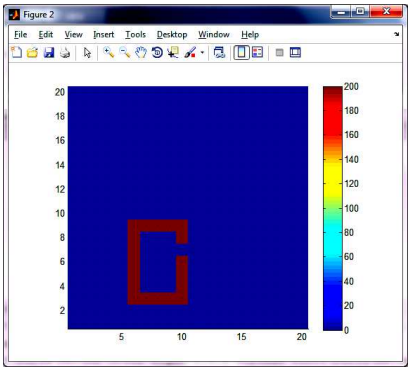

Рис.1. Відображення введених початкових умов

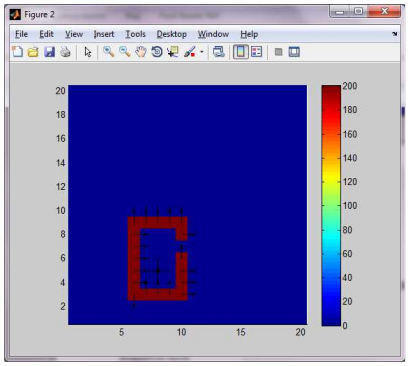

Рис.2. Результат першої ітерації

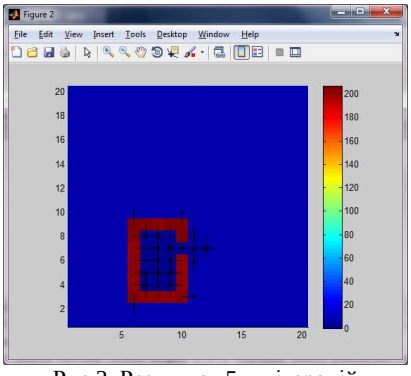

Рис.3. Результат 5-ти ітерацій

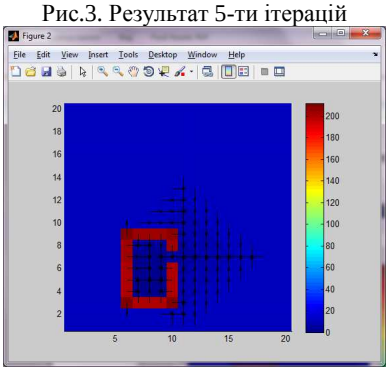

Рис.4. Результат 10-ти ітерацій

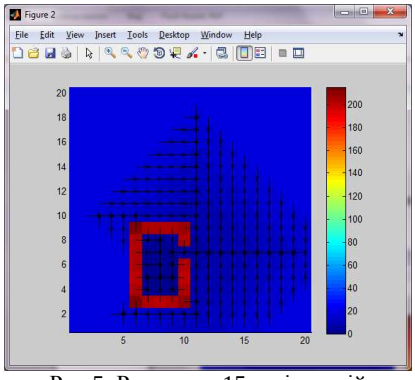

Рис.5. Результат 15-ти ітерацій

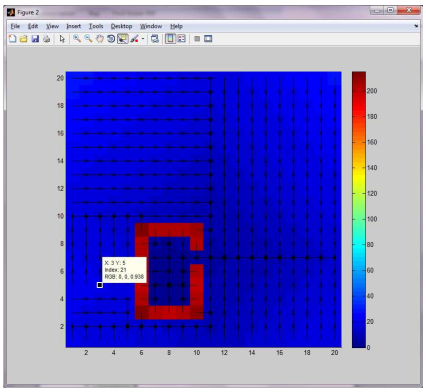

Рис.6. Кінцевий результат

Бачимо, що вартість переходу з необхідного початкового положення об'єкту складає 21 одиницю (21 крок і 4 повороту).

**Висновки**

Таким чином в запропонованій статті розглянуто основні особливості реалізації метода динамічного програмування для формування оптимального маршруту переміщення мобільного робота. Показано проведення експериментів формування оптимального маршруту у середовищі MatLab (на різних кроках виконання програмного забезпечення.

**СПИСОК ЛІТЕРАТУРИ:**

1. *MATLAB 7[Текст] / И.Е. Ануфриев, А.Б. Смирнов, Е.Н. Смирнова – Санкт-Петербург, «БХВ-Петербург», 2005р. – 1104с.*

2. *Методы оптимизации в теории управления: Учебное пособие / И. Г. Черноруцкий. — СПб.: Питер, 2004. — 256 с.*

3. *Webinar on topic “Optimization in MATLAB: An Introduction to Quadratic Programming” [Electronic resource] / The Mathworks – Access mode: www/URL:http://www.mathworks.com/company/events/webinars/wbnr62151.html?id=62151&p1=961663319&p2=961663337 .*

4. *Russ Tedrake. UNDERACTUATEDROBOTICS: Algorithms for Walking, Running, Swimming, Flying, and Manipulation CHAPTER 9. Dynamic Programming [Электронный ресурс] / Режим доступа: http://people.csail.mit.edu/russt/underactuated/underactuated.html?chapter6.*

5. *Алгоритмы обхода препятствий - [Электронный источник] – режим доступа http://pmg.org.ru/ai/navigato.htm.*