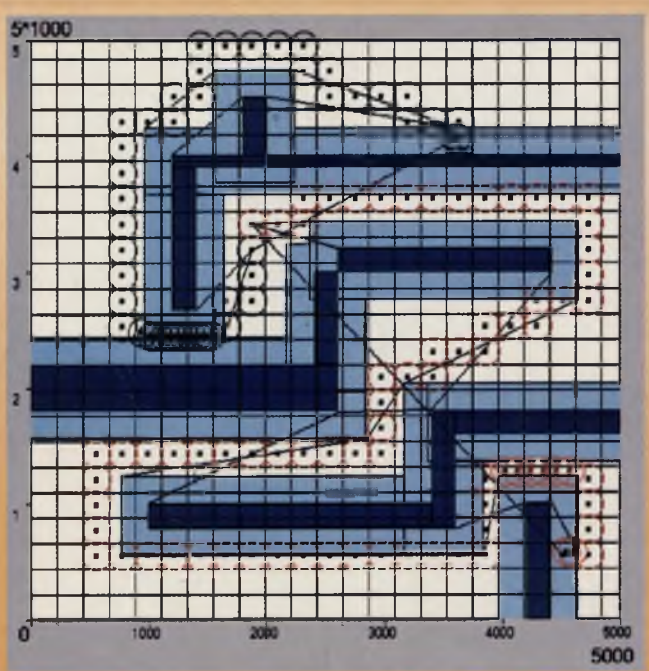
ZESZYTY NAUKOWE
POLITECHNIKI ŚLĄSKIEJ

P. 3361/04

Gabriel G. KOST

PLANOWANIE BEZKOLIZYJNYCH ŚCIEŻEK MANIPULACYJNYCH I STACJONARNYCH ROBOTÓW PRZEMYSŁOWYCH OPARTE NA PROCESACH DECYZYJNYCH MARKOWA I FUNKCJI OCEN



Gliwice 2004

POLITECHNIKA ŚLĄSKA
ZESZYTY NAUKOWE
Nr 1655

P. 3361/04

Gabriel G. KOST

Planowanie bezkolizyjnych ścieżek manipulacyjnych i stacjonarnych robotów przemysłowych oparte na procesach decyzyjnych Markowa i funkcji ocen

Gliwice 2004

Opiniodawcy

Prof. dr hab. inż. Jan AWREJCEWICZ
Prof. dr hab. inż. Edward CHLEBUS

Kolegium redakcyjne

Redaktor naczelny – Prof. dr hab. inż. Andrzej BUCHACZ
Redaktor działu – Dr hab. inż. Ryszard NOWOSIELSKI
Profesor nzw. w Politechnice Śląskiej
Sekretarz redakcji – Mgr Elżbieta LEŚKO

Redakcja

Mgr Anna BŁAŻKIEWICZ

Redakcja techniczna

Alicja NOWACKA

Wydano za zgodą
Rektora Politechniki Śląskiej

PL ISSN 0434-0817

© Copyright by
Gabriel G. KOST
Gliwice 2004

P. 130/04

Pracę tę dedykuję

żonie Halinie
oraz Oli i Martynie

Spis treści

str.

Wykaz ważniejszych oznaczeń 9

1. Wstęp 15

2. Zrobotyzowane systemy technologiczne 18

2.1. Środowisko technologiczne robota 18

2.1.1. Warunki organizacyjne sceny robota 19

2.1.2. Warunki konstrukcyjno-geometryczne (konfiguracja przestrzenna środowiska robota) 22

2.1.3. Warunki dynamiczne otoczenia robota manipulacyjnego 22

2.2. Programowanie robotów przemysłowych 26

3. Planowanie zadań robotów 34

3.1. Metody planowania trajektorii robotów 37

3.2. Analiza przestrzeni 40

3.2.1. Metody topograficzne 41

3.2.2. Metody geometryczne 43

3.2.2.1. Metody heurystyczne 45

3.2.2.1.1. Metoda drzew oktalnych 46

3.2.2.1.2. Kulowa aproksymacja przestrzeni 52

3.2.2.2. Metoda probabilistyczna 54

3.2.2.3. Metoda sztucznego potencjału 56

3.2.3. Inne metody planowania trajektorii w zewnętrznej przestrzeni zadania 58

3.3. Wnioski z literatury 59

4. Cel, zakres i teza pracy 64

4.1. Założenia metody 66

5. Metoda planowania bezkolizyjnych ścieżek dla robotów manipulacyjnych 68

5.1. Strategia działań robota 69

5.1.1. Własności metody uczenia się ze wzmocnieniem 71

5.1.2. Adaptacja metody uczenia się ze wzmocnieniem do zadania programowania robota 73

5.2. Przestrzeń wokółrobotowa 77

5.2.1. Przestrzeń robota manipulacyjnego ... 77
5.2.2. Ścieżka robota w przestrzeni stanu ... 81
5.2.3. Przeszkody – obiekty sceny ... 83
5.2.4. Przestrzeń robocza robota ... 83
5.2.4.1. Przestrzeń kolizyjna ... 83
5.2.4.2. Analiza przestrzeni ... 84
5.3. Procesy sterowania stochastycznego ... 91
5.3.1. Procesy decyzyjne Markowa i własność Markowa ... 92
5.3.1.1. Adaptacja procesu decyzyjnego Markowa do zadania planowania ścieżki robota ... 96
5.4. Planowanie ścieżki robota ... 98
5.4.1. Warunki zadania ... 99
5.4.2. Model PLANERA ścieżek bezkolizyjnych ... 101
5.4.3. Ocena podjętych działań ... 104
5.4.3.1. Funkcja prawdopodobieństwa osiągnięcia stanu s_i ... 106
5.4.3.2. Prawdopodobieństwo przejścia ... 107
5.4.3.3. Ocena podjętych działań ... 109
5.5. Dyskretna funkcja wag ... 111
5.5.1. Strefa oddziaływania obiektów sceny na robota ... 111
5.6. Strategia poszukiwania bezkolizyjnych ścieżek robota ... 116
5.6.1. Strategia ψ wyznaczania najkrótszej ścieżki robota ... 117
5.6.1.1. Strategia pomocnicza ψ_1 „prosto do celu" ... 118
5.6.1.2. Strategia pomocnicza ψ_2 „omiń" ... 120
5.7. Ograniczenia uzyskiwanych rozwiązań ... 123
5.8. Algorytm logiczny opracowanej metody ... 125
5.8.1. Wiz_SRM - symulator PLANERA bezkolizyjnych ścieżek robota ... 126
6. Weryfikacja metody. Eksperyment symulacyjny ... 130
6.1. Eksperyment symulacyjny ... 131
6.1.1. Dane ... 131
6.1.2. Wyniki symulacji ... 132
6.2. Ocena kolizyjności działań robota w systemach programowania off-line robotów 149
6.3. Wnioski ... 154
7. Podsumowanie ... 156
Literatura ... 163
Streszczenie ... 181

Table of Contents

Page
List of key terms ... 9
1. Introduction ... 15
2. Robotized technical systems ... 18
2.1. Technical environment of the robot ... 18
2.1.1. Organizational conditions of the operation scene of the robot ... 19
2.1.2. Structural and geometrical conditions (configuration of space) of the environment surrounding the robot ... 22
2.1.3. Dynamical conditions of the environment surrounding the manipulative robot ... 22
2.2. Programming of industrial robots ... 26
3. Planning the tasks of the robot ... 34
3.1. Methods of planning the robot trajectory ... 37
3.2. Analysis of space ... 40
3.2.1. Topographic methods ... 41
3.2.2. Geometric methods ... 43
3.2.2.1. Heuristic methods ... 45
3.2.2.1.1. Octal trees method ... 46
3.2.2.1.2. Spherical approximation of space ... 52
3.2.2.2. Probabilistic method ... 54
3.2.2.3. Artificial potential method ... 56
3.2.3. Other methods of planning robot trajectory in the external space of the task 58
3.3. Conclusions drawn from reference literature ... 59
4. Purpose, scope and thesis ... 64
4.1. Assumptions of the method ... 66
5. Method of planning collision free paths for manipulative robots ... 68
5.1. Strategy of the operations of the robot ... 69
5.1.1. Character of the reinforced learning method ... 71
5.1.2. Adjustment of the reinforced learning method to the task of robot programming ... 73

5.2. Space surrounding the robot 77
5.2.1. Space of the manipulative robot 77
5.2.2. Operation path of the robot in space 81
5.2.3. Obstacles- objects in the operation scene of the robot 83
5.2.4. Operation space of the robot 83
5.2.4.1. Collision space 83
5.2.4.2. Analysis of space 84
5.3. Processes of stochastic control 91
5.3.1. Markov's decision-making processes and their character 92
5.3.1.1. Adjustment of Markov's decision-making process to the task of planning the path of the robot 96
5.4. Planning the path of the robot 98
5.4.1. Conditions of the task 99
5.4.2. Model of collision-free path PLANNER 101
5.4.3. Assessment of the operations undertaken by the robot 104
5.4.3.1. Function of the probability of reaching state s_i 106
5.4.3.2. Probability of passage 107
5.4.3.3. Assessment of the operations undertaken by the robot 109
5.5. Discrete weights function 111
5.5.1. Zone of the impact of technical objects in the robot scene on the robot 111
5.6. Strategy of the search for collision-free paths of the robot 116
5.6.1. Strategy ψ of designating the shortest path of the robot 117
5.6.1.1. Supporting strategy ψ_1 "straight to the destination" 118
5.6.1.2. Supporting strategy ψ_2 "avoid" 120
5.7. Constraints to the solutions 123
5.8. Logical algorithm of the designed method 125
5.8.1. Wiz_SRM – simulator of the collision – free path PLANNER 126
6. Verification of the method. Simulation experiment 130
6.1. Simulation experiment 131
6.1.1. Data 131
6.1.2. Simulation results 132
6.2. Assessment of the collision capacity of the robot operations in the off-line programming systems 149
6.3. Conclusions 154
7. Recapitulation 156
References 163
Abstract 181

Wykaz ważniejszych oznaczeń

γ_{i+1} - współczynnik proporcjonalności zmiany stanu s_{i+1},

δ_i - funkcja przejścia określająca sposób zrealizowania kroku t_i zadania,

δ^2 - odległość P_{wi} od LG,

ψ - strategia lokalna wyznaczania najkrótszej drogi do celu,

ψ_1 - lokalna strategia „prosto do celu",

ψ_2 - lokalna strategia „omiń",

ω - funkcja wag,

ξ - ocena przyznana za dobrze zrealizowane zadanie, tj. wartość funkcji wzmocnienia,

ΔS^p - lokalny podzbiór p stanów,

Π_{RP} - przestrzeń robocza robota,

Π_w - przestrzeń bezkolizyjna,

Π_{SC} - przestrzeń sceny,

Π_{OB} - przestrzeń kolizji robota,

Γ - zadanie ruchu realizowane przez robot (bezkolizyjna ścieżka robota określona odpowiednią łamaną w przestrzeni, zadanie przejścia pomiędzy dwoma wyznaczonymi położeniami w przestrzeni: początkowym i końcowym) (21),

Ψ - ogólna strategia działania planera,

Ω - funkcja wag,

a_i - akcja podjęta przez robota,

b - długość boku oktantu (rastra),

b^0 - długość boku oktantu zerowego (dla $p=0$),

b^p - długość boku oktantu na poziomie p podziału oktalnego przestrzeni (dla $p>0$),

b^s - długość boku oktantu startowego,

$card(\cdot)$ - liczba kardynalna zbioru,

d_i - odległość pomiędzy dwoma sąsiednimi punktami węzłowymi planowanej ścieżki robota,

d_s - średnica przedmiotu manipulowanego,

$\vec{d}_{i+1}$ - wektor przejścia odpowiadający drodze d_{i+1},

$|\vec{d}_{i+1}|$ - odległość (d_i) pomiędzy dwoma sąsiednimi punktami węzłowymi w sensie euklidesowym,

g_κ^u - krawędź κ w prostopadłościanie u,

i - indeks dolny, określający element bieżący,

$i+1$ - indeks dolny, określający element kolejny względem i,

j - indeks górny: obiekt j sceny (przeszkoda),

k - horyzont planowania bezkolizyjnej ścieżki robota w przestrzeni,

l_1 - odległość robota od przeszkody,

l_2 - głębokość wejścia robota pomiędzy dwie sąsiadujące przeszkody,

l_{OB} - długość najkrótszego boku sąsiadujących przeszkód,

p - poziom podziału oktalnego przestrzeni sceny robota,

$p(\cdot)$ - prawdopodobieństwo zajścia zdarzenia,

$p(s_i)$ - prawdopodobieństwo zajścia (osiągnięcia) stanu s_i,

$p(s_i,s_{i+1})$ - prawdopodobieństwo przejścia pomiędzy stanami s_i i s_{i+1},

q_i - wewnętrzna konfiguracja i manipulatora robota (wektor współrzędnych uogólnionych) określona w układzie stanów wewnętrznych,

$r(\cdot)$ - funkcja oceny,

r_u - ruchliwość łańcucha kinematycznego manipulatora,

s_i - położenie i (stan i konfiguracji przestrzennej) robota w przestrzeni zadania,

s_1 - stan związany z położeniem początkowym planowanej ścieżki robota,

s_k - stan związany z położeniem końca planowanej ścieżki robota,

${}_s s_i$ - stan sąsiedni względem stanu s_i,

s_i^p - stan i związany z p-oktantem,

$s_i(P_{wi})$ - bieżący stan s_i robota osiągnięty w punkcie węzłowym P_{wi},

t_i - kolejny krok robota w przestrzeni wyznaczony przez planer wzdłuż wyznaczonej ścieżki pomiędzy dwoma sąsiednimi punktami węzłowymi,

A^* - algorytm przeszukiwania heurystycznego drzew decyzyjnych,

A^Ψ - podzbiór zbioru A określony strategią Ψ,

$A(a_i)$ - zbiór akcji a_i,

BFS - heurystyczny algorytm przeszukiwania „wszerz",

C_0 - waga zdobywanych nagród związana z obszarem nie podlegającym wpływowi V_{OB},

C_1 - waga zdobywanych nagród związana ze strefą oddziaływania obiektu V_{OB},

C_1^j - współczynnik korekcyjny wagi C_1,

C_2 - waga związana ze wzajemną bliskością przeszkód,

C_3 - waga kształtująca wysokość zdobywanej nagrody zależnie od przekroczenia C_T,

C_k - wartość korekcyjna oktantu startowego,

C_T - parametr określający szerokość „korytarza planowania",

$D_{sP_{wi+1}}$ - suma długości dróg prowadzących do punktów węzłowych sąsiednich z położenia bieżącego,

$E[\cdot\ ]$ - wyrażenie o uśrednionej wartości,

$G(g_{\kappa}^{u})$ - zbiór κ par przecinających się krawędzi g prostopadłościanu u,

$H(h_{g}^{u})$ - zbiór g wierzchołków h w prostopadłościanie u,

K^{p} - cecha p-oktantu,

L - wzajemna odległość dwóch sąsiadujących obiektów,

L_{PM} - długość przedmiotu manipulowanego,

LG - linia geodezyjna,

M - macierz przejść procesu decyzyjnego Markowa,

N^{0} - liczność zbioru oktantów na poziomie $p=0$,

N^{p} - liczność zbioru $S(s_i)$ na poziomie p,

N^{s} - liczba rastrów (liczność zbioru stanów) poziomu startowego,

N_s - liczność zbioru S,

N_V - liczebność zbioru V poszukiwań heurystycznych (liczba wierzchołków drzewa decyzyjnego),

N_w - liczba (liczność zbioru) rastrów należących do przestrzeni bezkolizyjnej,

${}_sN$ - liczność zbioru stanów sąsiednich ${}_sS$,

N_w^{p} - liczba rastrów poziomu p należąca do przestrzeni bezkolizyjnej,

${}_sN^{p}$ - liczność zbioru stanów sąsiednich utworzonych na poziomie p,

OB^{j} - obiekt j sceny (przeszkoda robota),

P_l - położenie początku planowanej ścieżki, $P_l=P_{wl}$,

P_k - położenie końca planowanej ścieżki, $P_k = P_{wk}$,

P_{wi} - punkt węzłowy i ścieżki robota,

PR^{u} - prostopadłościan u,

$S(s_i)$ - przestrzeń stanów zewnętrznych robota zdefiniowana w przestrzeni zadania,

S^{Ψ} - podzbiór zbioru S określony strategią Ψ,

${}_sS$ - zbiór stanów sąsiednich,

T_Q - transformacja prosta kinematyki robota,

$T(\theta_e^{u})$ - zbiór e ścian θ prostopadłościanu u,

U_{RP} - układ współrzędnych robota,

U_{SC} - podstawowy układ współrzędnych kartezjańskich sceny robota,

V_{OB} - strefa oddziaływania obiektu (przeszkody) na otoczenie,

V - graf-drzewo decyzyjne metody oktalnej,

Q - wewnętrzna przestrzeń konfiguracyjna robota.

1. Wstęp

Programowanie robotów przemysłowych jest jednym z najbardziej czasochłonnych, złożonych i trudnych etapów technologicznego przygotowania produkcji (TPP [24,26,28]). Obecnie procesy TPP w większości przypadków realizowane są komputerowo, gdyż wymuszana warunkami pracy integracja współczesnych systemów technologicznych zachodzi głównie na płaszczyźnie informatycznej i informacyjnej [23,24,25,89,94,113,176]. Działania te, przy zachowaniu warunku kompatybilności danych, pozwalają na osiągnięcie integracji informacyjnej oraz organizacyjnej składników systemu technologicznego. W każdym, komputerowo i organizacyjnie zintegrowanym systemie technologicznym, kluczowym elementem integrującym, obok systemu informatycznego, jest system transportowy (lub transportowo-manipulacyjny) [31,64,90,95,106,152,166,167]. Algorytm pracy systemu transportowego jest bowiem nośnikiem informacji o przebiegu (harmonogramie i logice) całego procesu technologicznego pomiędzy kolejnymi stanowiskami produkcyjnymi. Niestety, proces TPP obejmujący zadanie programowania systemu transportowego na przykład robota, a tym samym synchronizację czasową z obsługiwanym procesem technologicznym, uwzględniającą logikę (harmonogram) realizowanego procesu technologicznego, jest najsłabszym ogniwem TPP. Problem ten jest tym bardziej wyrazisty, że dla pozostałych uczestników zintegrowanego procesu technologicznego stopień zaawansowania systemów komputerowego wspomagania jest na obecnym etapie rozwoju dobry i dostatecznie skuteczny [5,31,96,118,120,152,166]. Dotyczy to zarówno systemów klasy CAD, zintegrowanych systemów CAD/CAM, jak również niektórych etapów projektowania technologicznego, np. narzędzi oprzyrządowania, technologii, oprogramowania układów sterowania numerycznego maszyn [31,51,152,166] i planowania zadań w nich realizowanych. Przyczyn tego stanu należy szukać w tym, że programowanie środków transportowych (a w szczególności robotów manipulacyjnych) nie poddaje się łatwo procesowi algorytmizacji ze względu na złożoność logiczną realizowanych przez niego zadań

obsługi. Dla robotów manipulacyjnych problem ten potęguje również: złożoność geometryczną kinematyki robota i przestrzeni, w której się on porusza, potencjalnie bardzo dużą, oraz różnorodność możliwych wariantów rozwiązań realizowanego zadania [87,171].

Podejmowane od wielu lat próby rozwiązywania zagadnień komputerowego wspomagania programowania robotów przemysłowych [13,39,53,59,91,96,143,161,208,209] koncentrują się w znacznej części na problemie planowania zadań robota [14,20,35,40,59,90, 91,96,115,135,143,157,169,171,174,178,206]. Zagadnienie to, obejmujące zadania planowania akcji i bezkolizyjnych trajektorii robota [12,13,22,36,38,70,78,110,111,114,133, 156,168,197] w złożonym geometrycznie środowisku technologicznym robota, może być rozwiązywane zarówno w wewnętrznej przestrzeni stanu robota [43,69,92,171], jak i w przestrzeni zewnętrznej, w której jest zdefiniowane jego zadanie obsługi systemu technologicznego (tzw. przestrzeń zadania) [43,69,142,171]. Planowanie zadań robota rozwiązywane w jego wewnętrznej przestrzeni stanu koncentruje się na procesie optymalnego sterowania robotem [40,45,110,111,123,140,142,165,170,217], zaś rozwiązywane w przestrzeni zadania jest bezpośrednio związane z procesem programowania robota i koncentruje się głównie na poszukiwaniu możliwie najlepszego (najkrótszego, najprostszego i bezkolizyjnego) przejścia robota w przestrzeni z jego położenia bieżącego, do końcowego [15,40,46,49,62,74,114,127,133,134,142,165,197].

Zadanie planowania akcji robota koncentruje się na wygenerowaniu logicznie spójnego i uporządkowanego zbioru sekwencji czynności robota prowadzących do wygenerowania w trybie automatycznym instrukcji programowych robota, zmierzających do obsłużenia, zgodnie z wymaganiami procesu technologicznego, kolejnego stanowiska [69,77,78, 90,95,132,156,206,208].

Zadanie wyznaczenia trajektorii bezkolizyjnej w zewnętrznej przestrzeni zadania prowadzi do zdefiniowania ruchu robota w przestrzeni, opisanej kartezjańskim układem współrzędnych, przez utworzenie ciągu kolejnych położeń (miejsc) efektora robota w przestrzeni (dane geometryczne) tworzących jego ścieżkę, w oparciu o który tworzony jest tzw. zbiór rozwiązań dopuszczalnych. Miejsca te określają kolejne konfiguracje przestrzenne robota pozwalające na zrealizowanie jego przejścia z położenia początkowego do końcowego wzdłuż poszukiwanej bezkolizyjnej ścieżki. Dopiero opisanie znalezionej ścieżki bezkolizyjnej odpowiednią funkcją względem czasu pozwala zdefiniować trajektorię robota. Z tego powodu, zadanie planowania bezkolizyjnych ścieżek robota jest podstawowym składnikiem zadania planowania trajektorii robota, a to z kolei poprzedza zadanie sterowania ruchem robota, gdyż zbiór danych geometrycznych stanowi dane wejściowe do etapu

rozwiązania zadania optymalnego sterowania robotem [3,35,42,43,55,58,69,102,110,124, 141,142,164,165,166,170]. Zatem podstawowym warunkiem, jaki należy spełnić w procesie planowania trajektorii, jest zapewnienie robotowi orientacji w złożonej geometrycznie zewnętrznej przestrzeni, obejmującej obsługiwany system technologiczny, w której wypełnia on swoje zadanie obsługi, pozwalającej na rozróżnienie w niej obszarów kolizji i tych miejsc, gdzie jego ruch jest możliwy. Dzięki temu możliwe jest wygenerowanie danych określających poszukiwaną, bezkolizyjną trajektorię robota.

Podejmowane i prezentowane w literaturze [15,37,43,48,57,75,104,110,163,203,205, 218] próby rozwiązania zagadnienia analizy przestrzeni, niezbędnej w omawianym zadaniu, charakteryzują się koniecznością stosowania czasochłonnych algorytmów analizy heurystycznej. Dlatego ich przydatność praktyczna, np. z punktu widzenia wykorzystania w systemach komputerowego programowania robotów manipulacyjnych, jest stosunkowo niewielka, szczególnie w kontekście możliwości programowania on-line. Z tych powodów rozwijanie istniejących i poszukiwanie nowych metod pozwalających na dostatecznie sprawną analizę przestrzeni, tak by mogła być wykorzystana w systemach komputerowego wspomagania programowania robotów przemysłowych do wyznaczania bezkolizyjnych ścieżek (i/lub trajektorii) robotów, wydaje się w pełni uzasadnione.

2. Zrobotyzowane systemy technologiczne

Jednym z elementów zrobotyzowanych, elastycznych systemów technologicznych jest robot przemysłowy wykorzystywany zarówno jako środek transportowy-manipulacyjny, jak i maszyna technologiczna [64,65,66,84,157]. Wszystkie warunki i potrzeby związane z procesem aplikacji robota przemysłowego do nowych zadań obsługi wynikają z zadań, jakie musi on podjąć w systemie produkcyjnym. Zadania te kształtują zarówno zasady technologiczne, jak i organizacyjne funkcjonowania tego systemu, a tym samym określają warunki pracy robota. O ile warunki technologiczne określone są potrzebami realizowanego i ściśle zdefiniowanego procesu technologicznego, o tyle warunki organizacyjne są wypadkową zarówno procesu technologicznego, jak i warunków organizacji przestrzennej tego systemu (układ przestrzenny stanowisk, ich wielkość itp.). Oznacza to, że jednym z istotnych elementów warunkujących przebieg zadania obsługi systemu technologicznego przez robota, gwarantujących pełne zaspokojenie w nim potrzeb przepływu materiałów, jest organizacja przestrzenna tego systemu. To zaś określone jest warunkami geometrycznej konfiguracji wszystkich obiektów technologicznych i pomocniczych, składających się na wyposażenie technologiczno-organizacyjne zrobotyzowanego systemu technologicznego, takie jak: obrabiarki, magazyny wejściowe i wyjściowe, bufory przystanowiskowe, stanowiska reorientacji itp., które wpływają bezpośrednio na kształtowanie – oprócz wymagań technologicznych – zachowania się robota w tym otoczeniu.

2.1. ŚRODOWISKO TECHNOLOGICZNE ROBOTA

Środowisko technologiczne robota jest strukturą organizacyjną dość złożoną. Stanowią ją wszystkie elementy wyposażenia technologicznego zrobotyzowanego systemu produkcyjnego, począwszy od urządzeń technologicznych, jakimi są obrabiarki i maszyny

technologiczne sterowane numerycznie, poprzez wszelkiego typu magazyny przedmiotów i elementów wyposażenia technologicznego (narzędzi, chwytaków, buforów przystanowiskowych, stanowisk reorientacji itp.). Układ przestrzenny tych elementów jest dość różnorodny, zależnie od przeznaczenia technologicznego danego zrobotyzowanego systemu produkcyjnego. To zaś zależy od rodzaju realizowanego w tym systemie procesu technologicznego (równowagi czasowej operacji technologicznych), typoszeregu produkowanych w nim przedmiotów oraz rodzaju i możliwości systemu transportowo-manipulacyjnego, w jaki jest on wyposażony (transporter liniowy, robot stacjonarny lub mobilny o ograniczonym zasięgu (torowisko), kilka robotów współpracujących) [84,86,157, 166,167]. Aktywność składników organizacyjnych systemu, oprócz warunków technologicznych, określona jest przyjętą zasadą sterowania tym systemem. W prostych organizacyjnie systemach zrobotyzowanych bardzo często układ sterowania robota spełnia, oprócz sterowania robotem, również rolę nadrzędnego układu sterowania tym systemem [6, 84,86]. W systemach bardziej zaawansowanych stosuje się sterowanie typu DNC [51,84, 118]). W obu przypadkach podstawą algorytmu sterowania systemem są przyjęte i zastosowane reguły organizacyjne systemu produkcyjnego, np. maksimum wydajności [51,166,167].

Określone powyżej podstawowe czynniki organizacyjne, wpływające na zachowanie się robota w zrobotyzowanym systemie produkcyjnym, określają aktywność i sposób zachowania się robota oraz częstotliwość obsługi poszczególnych stanowisk systemu produkcyjnego, a rodzaj realizowanego w nim procesu technologicznego określa specyfikę procesu obsługi stanowisk przez robota.

2.1.1. Warunki organizacyjne sceny robota

Środowisko technologiczne robota, które w zasadniczy sposób wpływa i kształtuje sposób rozwiązania zadania obsługi przez robota i jego zachowania się w przestrzeni, określane jest mianem sceny [91,113,132,157,205]. Scena obejmuje swoim zasięgiem wszystkie obiekty technologiczne i pomocnicze wykorzystywane w realizowanym procesie technologicznym i obsługiwane przez niego. Jest ona zatem kształtowana przez te obiekty, ale również przez samego robota, gdyż jego możliwości ruchowe definiują tzw. przestrzeń roboczą główną i pomocniczą, czyli obszar przestrzeni sceny, jaki może on penetrować, nie wyposażony, bądź wyposażony w odpowiednie narzędzie technologiczne [91,132,157,205]. Za każdym jednak razem należy w sposób jednoznaczny stwierdzić, że zgodnie z racjonalnie zdefiniowanymi zasadami organizacji systemu produkcyjnego [166,167] w scenie nie

występują żadne zbędne obiekty, czyli takie, których obecność nie wynika z realizowanego procesu technologicznego i tym samym których on nie obsługuje, a które przez to w jednoznaczny sposób byłyby wyłącznie przeszkodami w jego ruchu określonym zgodnie z algorytmem obsługi systemu.

Jednak, przyjmowane najczęściej w sterowaniu systemami produkcyjnymi kryterium maksymalnej wydajności produkcji oznacza, że sekwencja obsługi stanowisk przez robota nie pokrywa się w prosty sposób z chronologią operacji technologicznych. Najczęściej algorytm obsługi stanowisk systemu produkcyjnego budowany jest w oparciu o zasadę „przywoływania robota" [64,84,89] przez stanowisko technologiczne (obróbkowe), które tej obsługi wymaga, np. w celu pobrania obrobionego przedmiotu, lub założenia kolejnego itp. Oznacza to, że robot porusza się w scenie pomiędzy obiektami, zgodnie z pewnym – wynikającym z przebiegu procesu technologicznego – prawdopodobieństwem zajścia zdarzenia końca obróbki na jakimś stanowisku, a tym samym wystąpienia potrzeby jego obsługi i przywołanie robota. Zdarzenie to może zajść w dowolnej chwili, po wykonaniu wszystkich przewidzianych procesem zabiegów obróbkowych. Zasada ta sprawia, że koniec operacji obróbkowej na jednym stanowisku może zajść zarówno wtedy, kiedy robot oczekuje bezczynnie w swoim położeniu spoczynkowym, jak i wtedy, kiedy obsługuje on inne stanowiska. Nierównomierność - nawet niewielka - rozkładu czasów obróbki i czynności pomocniczych (zrównoważenie czasowe systemu) może spowodować, przy cykliczności obsługi, że kolejka zgłoszeń stanowisk oczekujących na obsługę może się tworzyć każdorazowo wg innej sekwencji. Oznacza to, że za każdym razem może dojść do sytuacji, w której do zgłaszającego się stanowiska robot musi przejść z innego miejsca sceny, realizując tym samym inną ścieżkę przejścia i omijając inne obiekty, które w tym konkretnym przypadku stanowią przeszkody na jego drodze.

W racjonalnie zorganizowanej i zaprojektowanej scenie, jak już wskazano, nie występują obiekty, których obecność nie byłaby technologicznie uzasadniona. To jednak nie zmienia faktu, że rola obiektów sceny jest dwojaka. Z jednej strony, w określonych przypadkach każde ze stanowisk jest obiektem docelowym działań robota i wtedy podlega obsłudze zgodnie z warunkami realizowanego procesu technologicznego; z drugiej zaś są one wyłącznie przeszkodami dla robota, zmierzającego do zrealizowania zaplanowanej technologią operacji obsługi innego stanowiska. Sytuacja ta sprawia, że w rzeczywistych warunkach sterowania robotem konieczne jest precyzyjne planowanie wszystkich sekwencji działań robota w przestrzeni. Stąd też bardzo ważnym elementem procesu programowania robota jest tzw. cyklogram [63,64,65,66,84,86,90,95]. Stanowi on uporządkowany w czasie

plan (harmonogram [143,144]) obsługi wszystkich stanowisk technologicznych zrobotyzowanego systemu produkcyjnego wg kolejności przewidywanych zgłoszeń, ustalonej w oparciu o znormowanie czasów pracy wszystkich stanowisk i czynności (ruchów) robota. O ile ustalenie normy czasowej operacji technologicznych nie stanowi większego problemu, o tyle określenie odpowiedniej normy czasowej czynności obsługi i przejść robota z położenia bieżącego do końcowego, związanego ze stanowiskiem przywołującym robota, jest dość istotnym problemem. Czasy są kształtowane w zasadzie przez wszystkie czynniki technologiczne realizowanego procesu i, co najważniejsze, w bardzo szerokim zakresie również przez intuicję i wyobraźnię programisty robota. Podstawowym kryterium jest bowiem przyjęta przez niego a priori prędkość ruchu robota, rozumiana jako prędkość przemieszczenia się efektora robota w przestrzeni po zadanej przez programistę ścieżce. Jak wiadomo, w sposób bardzo istotny wynika ona z warunków konstrukcyjno-technologicznych realizowanego procesu: wielkości (masa) przedmiotu manipulowanego i oprzyrządowania technologicznego robota, a także przewidywanej złożoności geometrycznej planowanej ścieżki robota, spowodowanej koniecznością omijania napotkanych przeszkód. O ile przenoszone przez robota masy są łatwe do dokładnego zmierzenia, o tyle przyjęta przez programistę prędkość robota jest parametrem ruchu czysto uznaniowym. Ta zaś kształtuje warunki dynamiczne ruchu robota [41,43,86,110,152]. Zatem czas planowanego przejścia robota pomiędzy określonymi w cyklogramie stanowiskami jest wartością szacunkową (często podlega pomiarowi podczas programowania i testowania gotowego programu robota). W skrajnych przypadkach, przy nie dość dokładnie określonym cyklogramie i wielokrotnie powtarzanych cyklach obsługi stanowisk przez robota, może doprowadzić do jego zaburzenia i załamania chronologii realizowanych operacji obsługi, wyznaczonych procesem technologicznym.

Z opisanej sytuacji można wyciągnąć pewne wnioski, dotyczące warunków środowiska technologicznego, w jakich działa robot. Mianowicie:

- środowisko technologiczne robota (scena) jest środowiskiem o dokładnie zdefiniowanej strukturze organizacyjnej, w której liczba obiektów technologicznych i pomocniczych jest ściśle i jednoznacznie określona,
- sposób przejścia robota pomiędzy poszczególnymi stanowiskami wynika z wystąpienia zdarzenia „potrzeby obsługi" zgłaszanej przez to stanowisko i kształtowana jest wg tej reguły,
- wszystkie obiekty sceny spełniają względem robota podwójną rolę: są zarówno obiektami obsługiwanymi przez niego, a równocześnie - dla innych sekwencji planowanego ruchu -

stanowią przeszkodę w jego swobodnym przemieszczaniu się w przestrzeni. Wymusza to konieczność ich omijania, a tym samym „modyfikuje" kształt drogi przejścia robota z jednego miejsca do drugiego.

Omawiane warunki występujące w scenie robota tworzonej przez obsługiwany przez niego system technologiczny nie uwzględniają warunków geometrycznych określających rozmieszczenie obiektów technologicznych sceny i bardzo istotnego czynnika, jakim jest jej dynamika.

2.1.2. Warunki konstrukcyjno-geometryczne (konfiguracja przestrzenna środowiska robota)

Jak już wspomniano, scena robota charakteryzuje się ścisłym ustabilizowaniem liczby obiektów, które się w niej znajdują, określonej zakresem i warunkami realizacji obsługiwanego procesu technologicznego. Oczywisty jest fakt, że skoro wszystkie obiekty sceny robota spełniają w realizowanym zrobotyzowanym systemie technologicznym rolę przypisaną im przez obsługiwany proces technologiczny, a robot pełni w nim rolę systemu transportowo-manipulacyjnego, to musi on mieć dostęp do każdego z tych obiektów. Warunkiem takiego dostępu jest spełnienie wymagania, by znajdowały się one w przestrzeni programowania [64,65], obejmującej główną i pomocniczą przestrzeń roboczą robota. Efektem konieczności spełnienia tego wymogu jest stosunkowo duże zagęszczenie obiektów w bliskim otoczeniu robota. Prowadzi to do tego, że swobodne poruszanie robotem jest znacząco utrudnione i wymaga ciągłej kontroli kolizyjności każdego jego ruchu względem obiektów sceny.

2.1.3. Warunki dynamiczne otoczenia robota manipulacyjnego

Z przeprowadzonych rozważań i analizy literatury robotycznej [12,46,64,65,69,75,110, 163,203] można wysnuć oczywisty wniosek, że dla robotów manipulacyjnych scena jest (powinna być) statyczna. Statyczność sceny oznacza, że jej konfiguracja przestrzenna, czyli rozlokowanie poszczególnych obiektów, nie ulega w trakcie ruchu robota żadnym zmianom. Stan ten wynika ze spełnienia zasady mówiącej o tym, że ze względu na trudności wykorzystania elementów dynamicznego śledzenia zmian zachodzących w otoczeniu robota, należy stworzyć robotowi warunki środowiska statycznego, w którym wszystkie jego elementy mają jednoznacznie określone miejsce i postać i są niezmienne w czasie lub zmiany te są niewielkie, ściśle określone, przewidywalne i nie wpływają wprost na ruch robota [135,137]. Zadaniem obsługi technicznej systemu zrobotyzowanego na etapie TPP jest więc

zapewnienie scenie pełnych warunków statyczności i ujęcie informacji o tym w cyklogramie, a tym samym w algorytmie programu robota. Jednak wymogi statyczności sceny nie obowiązują i nie mogą obowiązywać w sposób bezwzględny. Podstawowym powodem jest poruszający się w przestrzeni robot, który przenosi z obiektu na obiekt przedmioty, manipuluje nimi, zmienia swoje oprzyrządowanie technologiczne [10,66,69,84,110,135,137, 218]. Jednocześnie należy jednoznacznie stwierdzić, że robot dokonuje tych zmian okresowo, w ściśle określonym czasie i miejscu, w sposób przewidziany algorytmem jego programu [6,152,202,199]. Z tych powodów środowisko, mimo znaczących cech statyczności, takie jednak nie jest i czasami nazywane jest pseudostatycznym [91,135,156]. Czynniki te sprawiają, że tylko dzięki staraniom konstruktorów, obsługi technicznej i programistów spełniony zostaje podstawowy warunek, jakim jest statyczność sceny robota. Dzięki temu robot nie wymaga odpowiednio zaawansowanych systemów analizy sceny, a tym samym proces sterowania nim jest stosunkowo prosty, zaś dynamika otoczenia robota nie wpływa bezpośrednio na proces jego programowania.

Można jednak stwierdzić, że omówiony problem statyczności sceny robota ustępuje obecnie coraz bardziej na rzecz sceny dynamicznej, a mówiąc precyzyjnie - na rzecz takiej, w której problem jej dynamiki jest coraz bardziej widoczny i znaczący. Omówione szeroko środowisko statyczne występuje obecnie tak wyraźnie tylko w tych przypadkach systemów zrobotyzowanych, w których zastosowany robot przemysłowy przeznaczony jest wyłącznie do wykonywania typowych działań manipulacyjnych, związanych z algorytmem: „pobierz-przenieś-odłóż" (zadanie typu „pick-and-place" [59,77,132,135,156,211]). Jednak warunki i oczekiwania współczesnego przemysłu wykraczają znacząco poza tak oczywisty i prosty schemat. Rozwój systemów sterowania i oprogramowania robota, środków automatyzacji i coraz bardziej akcentowany wymóg elastyczności produkcji sprawiają, że roboty przemysłowe wykorzystywane są w coraz bardziej zaawansowanych organizacyjnie, technicznie i technologicznie systemach produkcyjnych. To, co do niedawna było wyłącznie domeną robotyki mobilnej [17,33,35,38,42,60,83,111,119,121,126,128,190,192,213,220], czyli dynamiczna zmiana otoczenia robota, staje się coraz większym udziałem stacjonarnych i manipulacyjnych robotów przemysłowych [111,197,206,215,220]. Dla robotów mobilnych dynamika jego otoczenia wynika z faktu, że porusza się on swobodnie pomiędzy obiektami znajdującymi się w jego otoczeniu. By uniknąć kolizji, wyposażany jest w złożony system postrzegania sceny, śledzenia zmian, jakie w nim zachodzą, i stosownie do nich modyfikuje swój algorytm postępowania, dobierając odpowiednią sekwencję swoich ruchów, tak by zrealizować postawione zadanie i jednocześnie zrobić to bezkolizyjnie. Obecnie zaś podobne

problemy zaczęły pojawiać się również w odniesieniu do robotów stacjonarnych [36,111,164,165,206,215,220]. Wymagane zmiany dynamiczne zachodzące w geometrycznej konfiguracji przestrzennej systemu technologicznego, wymuszane realizowanym procesem, czy pojawienie się drugiego robota i konieczność współpracy pomiędzy nimi powodują, że coraz trudniej jest spełnić warunki statyczności sceny. Do zilustrowania tej tezy przytoczone zostaną dwa przykłady:

1) elastyczna linia produkcyjna (elastyczne gniazdo produkcyjne), w której robot (lub kilka robotów) współpracuje z przenośnikiem (lub wózkiem) zmieniającym cyklicznie położenie przedmiotów manipulowanych, co dzieje się równolegle, w trakcie realizowania przez niego działań obsługi (np. moduł produkcyjny lub gniazdo – na linii montażowej lub lakierniczej karoserii samochodowych);
2) dwa roboty manipulacyjne i stacjonarne (lub kilka robotów) zainstalowane na wózku, sterowanym, jako kolejna oś robota, mające wspólny obszar przestrzeni roboczej.

W obu przypadkach możliwość pojawienie się dodatkowego obiektu (o stosunkowo dużych gabarytach) wymusza zmianę zachowania się robota: z jednej strony konieczność obsługi zgłaszającego się obiektu, z drugiej zaś konieczność ominięcia pojawiającej się bez udziału robota przeszkody. Przypadki te są o tyle ciekawe, że pojawienie się dodatkowego obiektu w przestrzeni robota (np. kolejna blacha poszycia montowanej karoserii samochodu, bądź ramię drugiego robota) powoduje drastyczne ograniczenie dostępnej dla robota przestrzeni. Na liniach produkcyjnych, na których zarządzanie i starowanie programem produkcji odbywa się komputerowo, dąży się do tego, by asortyment produkowanych wyrobów mógł ulegać ciągłym i płynnym zmianom (produkcja elastyczna), co powoduje, że układ przestrzenny sceny robota może się zmieniać nieregularnie. Stan taki wymusza konieczność dostosowania zachowania robota do aktualnej konfiguracji geometrycznej sceny. Oczywiście, staraniem programistów jest, by zmiany otoczenia robota były poddane zasadniczym rygorom: zachodziły regularnie, w ściśle określonej chronologii, co pozwala je związać z odpowiednią, przygotowaną zawczasu sekwencją programu robota. Warunki te, podobnie jak dla sceny statycznej, wymuszają zachowanie szczególnej dbałości o uporządkowanie logiczne algorytmu robota i sekwencji ruchów, by spełnić podstawowy warunek, jakim jest praca bezkolizyjna. Jakiekolwiek zachwianie tego porządku powoduje nieprzewidziane zaburzenie pracy robota i najczęściej prowadzi do zatrzymania pracy systemu (a nawet awarii spowodowanej kolizją robota z otoczeniem).

Omówione przypadki środowisk robotów stacjonarnych, w których problem ich dynamiki istotnie wpływa na zachowanie się robota, powinny wymusić w konsekwencji

również zmianę zasad zachowania się robota w takim otoczeniu ze sztywnego, niezmiennego na bardziej elastyczne, w którym robot potrafiłby zaadaptować swoje zachowanie odpowiednio do wykrytych zmian konfiguracji przestrzennej tego otoczenia. Mechanizm taki mógłby uprościć, w sferze organizacyjnej, jak i algorytmicznej, programowanie robotów do pracy w systemach elastycznych, zwolnić programistę robota z rygorów przewidywania i przygotowywania zawczasu sekwencji programu robota, zawierających trajektorie obejściowe, dostosowane do wariantów lokalnych zmian konfiguracji przestrzennej sceny.

Z doświadczeń programistów pracujących na liniach montażowych FIATA (COMAU Poland Sp. z o.o.), czy Mercedesa, Audi i BMW (LCD Technologies, Szwajcaria) wynika, że cykl dostosowania programu robota przygotowanego techniką off-line (program oraz dodatkowe osensorowanie środowiska [31,44,81,82,117,153,154,183,189,194], pozwalające spełnić warunek jego statyczności) dla jednego gniazda montażowego na linii produkcyjnej wynosi ok. 4 tygodnie. Zaś samo napisanie programu techniką off-line (np. systemy: IGRIP, eMPower – dawniej Robcad, Robsim, czy COSIMIR) trwa od 3 do 4 tygodni (na samej linii montażowej gotowy off-line program robota jest tylko doprecyzowywany i zsynchronizowany z otoczeniem). Dlatego też, mimo stosowania zaawansowanych systemów programowania off-line robotów (np. COSIMIR, IGRIP, eMPower, itp.), tak podstawowe problemy, jak i wybór ścieżki robota w przestrzeni i ocena jej kolizyjności z otoczeniem wymaga dalej weryfikacji w warunkach rzeczywistego środowiska robota i oceny wizualnej jego zachowania.

Rozwiązaniu tego zadania mogłaby posłużyć metoda wyznaczania bezkolizyjnych przejść robota w przestrzeni, w oparciu o prostą informację pochodzącą z systemu sensorowego współpracującego z robotem, a wskazującą obecność nowego (zdefiniowanego) elementu w otoczeniu robota. Metoda taka dałaby możliwość elastycznego zareagowania na zachodzące lokalnie zmiany w jego scenie bez konieczności przebudowy algorytmu programu sterującego robotem. Zwolniłoby to programistę z konieczności precyzyjnego i sztywnego konfigurowania cyklogramu programu robota i dało układowi sterowania robotem możliwość elastycznego reagowania na zachodzące zmiany bez konieczności przewidywania i przygotowywania „zawczasu" wszystkich możliwych reakcji robota i sekwencji ruchów (dróg obejść) napotkanych przeszkód. Stąd też problem planowania działań robota, a w szczególności bezkolizyjnych trajektorii w przestrzeni zadania (sceny), który pozwoliłby na wspomaganie metody programowania off-line robotów stacjonarnych i manipulacyjnych, jest obecnie coraz bardziej ważny.

2.2. PROGRAMOWANIE ROBOTÓW PRZEMYSŁOWYCH

Robot przemysłowy, jako maszyna przeznaczona do zastępowania działań człowieka w zautomatyzowanych systemach technologicznych, powinien być zdolny do wypełniania wszystkich tych czynności, które wykonuje obsługa takiego systemu. Tym samym powinien mieć zdolność zapewnienia ciągłości jego pracy w określonym, przewidzianym warunkami organizacyjno - technologicznymi, czasie. Wynika z tego, że sposób programowania robota (metoda) powinien umożliwiać prawidłowe zrealizowanie wszystkich tych zadań w możliwie najlepszy sposób. Warunki współczesnej organizacji produkcji są coraz dalsze od warunków pracy ustabilizowanej, jakie do niedawna były łączone z produkcją średnio- i wielkoseryjną. Coraz częściej wymaga się, by system produkcyjny był zdolny do zaspokojenia szybko zmieniających się potrzeb określonych wymogami rynku, czego wyraźnym skutkiem jest wymóg elastyczności produkcji. Sawik podaje [166], że elastyczność w ogólnym znaczeniu jest wypadkową wszystkich składników procesu produkcyjnego, tak organizacyjnych, jak technicznych i technologicznych oraz zautomatyzowanych i komputerowych systemów wspomagających te działania. Warunkiem zatem efektywnego wykorzystania robota w wysoko zintegrowanych systemach wytwarzania jest nie tylko jego uniwersalność kinematyczna, ale również pełna automatyzacja jego prac aplikacyjnych. W tej grupie zadań jednym z najważniejszych jest zadanie zautomatyzowanego programowania robota do realizacji zadań w wysoko zintegrowanym, zautomatyzowanym środowisku technologicznym. Warunkiem właściwego wykorzystania takich zrobotyzowanych systemów technologicznych jest ich wysoka sprawność produkcyjna (wydajność i elastyczność), która wynika m. in. z nieprzerwanej pracy systemu. Ta zaś wymusza konieczność wprowadzania do jego obsługi wszelkich możliwych zautomatyzowanych środków wspomagania produkcji na poziomie: organizacji, planowania zadań oraz programowania. Ostatnią grupę środków wspomagających pracę zrobotyzowanych systemów wytwarzania stanowią komputerowe systemy programowania maszyn technologicznych. O ile w grupie komputerowych systemów programowania istnieje wiele systemów programowania obrabiarek sterowanych numerycznie (OSN), o tyle systemy programowania robotów przemysłowych właściwie nie są stosowane powszechnie. Tym samym najbardziej zawodnym składnikiem są wszelkie systemy automatyzacji (komputeryzacji) prac wspomagających programowanie robotów przemysłowych [28,84,102,113,135,157]. Obecnie roboty przemysłowe są programowane najczęściej ręcznie metodą samouczenia (czasami proces ten jest w początkowej fazie wspomagany technikami off-line) [39,70,115,135,143,157,208,209,217]. Metoda ta wymaga

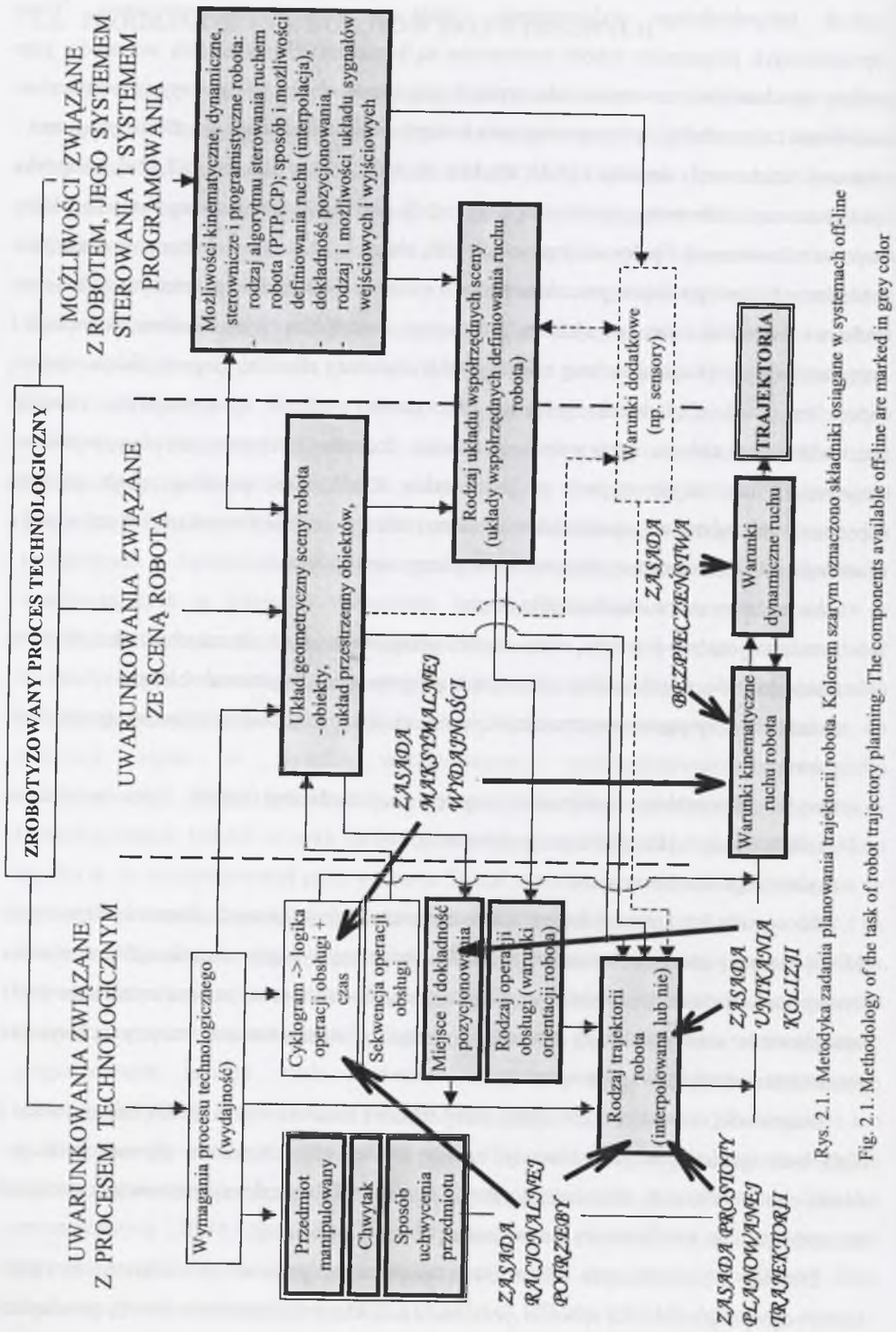
jednak bezpośredniego wykorzystania robota w procesie programowania. Testy opracowanych programów robota wymuszone są koniecznością określenia warunków jego pełnej synchronizacji z otoczeniem technologicznym, najczęściej z przyjętym kryterium minimum czasu obsługi, oraz sprawdzenia kolizyjności zdefiniowanego off-line programu i dynamiki zachowania się robota [7,43,102,110,132,138,139,184,185,186,187,188]. Metodyka programowania robota manipulacyjnego (rys. 2.1) pokazuje złożoność tego procesu, która wynika z konieczności połączenia ze sobą trzech różnych jakościowo elementów: warunków określonych obsługiwanym procesem technologicznym, warunków geometrycznych sceny robota i możliwości samego systemu robotowego (manipulator, jego system sterowania i programowania). Obejmuje szereg etapów, z których każdy charakteryzuje się swoją odrębną specyfiką i określa charakterystyczne dla siebie warunki – wymagania, stawiane prowadzonemu zadaniu programowania robota, i często jest uwarunkowany wynikami uzyskanymi na innych etapach programowania. Dodatkowo, przebieg całego procesu programowani robota w systemie produkcyjnym musi być poddany weryfikacji wynikającej z konieczności stosowania racjonalnych zasad postępowania. Są to:

- zasada planowania ruchu bezkolizyjnego,
- zasada racjonalnej potrzeby, wymuszająca podejmowanie jedynie takich działań, które są niezbędne do prawidłowego zrealizowania programowanego procesu obsługi,
- zasada prostoty planowanej trajektorii, wymuszająca poszukiwanie jedynie najprostszych rozwiązań,
- zasada maksymalnej wydajności programowanych działań robota, która wynika ze zdefiniowanego cyklogramu programu robota,
- zasada bezpieczeństwa pracy.

Na rysunki 2.1 kolorem szarym zaznaczono te składniki procesu planowania trajektorii robota, które są dostępne w znanych na rynku systemach programowania off-line robotów przemysłowych. Niektóre z nich pozwalają na modelowanie sceny robota technikami CAD (modelowanie sceny: obiekty, chwytaki, przedmioty manipulowane, miejsca uchwycenia przedmiotów manipulowanych itp.).

Oczywiście, warunki bezkolizyjnej pracy robota i bezpieczeństwa muszą być spełnione i mają bezwzględny priorytet. Niestety, zasady te nie mają charakteru sformalizowanego. Pełnią one jedynie rolę pomocniczą, jako zalecenia, których zakres zastosowania i stopień ważności podlega każdorazowo indywidualnej ocenie programisty.

Efektem podejmowanych od ponad trzydziestu lat prób w tym zakresie są języki tekstowego programowania robotów poziomu manipulacji (manipulation level), przedmiotu



Rys. 2.1. Metodyka zadania planowania trajektorii robota. Kolorem szarym oznaczono składniki osiągane w systemach off-line

Fig. 2.1. Methodology of the task of robot trajectory planning. The components available off-line are marked in grey color

manipulowanego (object level) i zadania (task level) [13,20,39,59,102,113,135,209,217], a obecnie metody programowania off-line robotów oparte na zaawansowanej grafice komputerowej i systemach CAD [27,28,77,91,121,144,174,203,209,215,218]. O ile poziom manipulacji umożliwia jedynie programowanie ruchów robota i jest tożsamy z programowaniem samouczącym oraz wykorzystuje do tego celu sformalizowany język programowania wysokiego poziomu, o tyle języki poziomu object i task charakteryzują się w założeniu znacznie większą integracją programowanych czynności robota. Języki tych poziomów wykorzystują formalizm planowania zadań elementarnych robota [13,59,63,69,77,96,209]. Łączą w sobie proces planowania zarówno akcji, jak i trajektorii robota i oparte są na technikach sztucznej inteligencji [78,113,132,171,210,218].

Niestety, próby czynione w zakresie opracowania i aplikacji języków programowania robotów przemysłowych nie przyniosły spodziewanych rezultatów. Istniejąca grupa języków programowania robotów poziomu manipulacji nie spełniła w 100% związanych z nimi oczekiwań. Głównym problemem ich aplikacji (na poziomie manipulacji) stała się wymagana przestrzenna złożoność definiowania geometrycznego planowanego ruchu robota oraz częściowo wirtualny model sceny (tzw. model świata robota – World Model [12,14,91,93,113,135,157,172,174,175,200,201], w którym pracuje robot, a na poziomie „object" i „task" dodatkowo problemy z rozwiązaniem zadania planowania akcji.

Dla robota zagadnienie ruchu jest zagadnieniem zawsze wielowymiarowym. Zarówno wewnętrzna przestrzeń stanu robota (przestrzeń współrzędnych naturalnych), jak i przestrzeń zewnętrzna, opisująca scenę robota w kartezjańskim układzie współrzędnych, są przestrzeniami wielowymiarowymi. W obu przypadkach problem utrudnia dodatkowo względność wykonywanych przemieszczeń na poszczególnych osiach łańcucha kinematycznego robota [60,79,131].

Z zagadnieniem tym związany jest więc problem przełożenia geometrii 6-wymiarowej przestrzeni zadania robota (3 współrzędne kartezjańskie położenia i 3 kąty Eulera) na jego przestrzeń stanów wewnętrznych Q [43,45,102,126,162], wynikających z ruchliwości łańcucha kinematycznego manipulatora (najczęściej 5, rzadziej 4 lub 6 [65,66,69,132,137, 200,201]). Konieczność zastosowania mechanizmów kinematyki prostej i odwrotnej oraz wykorzystania do definiowania trajektorii robota notacji Denavita-Hartenberga (D-H) [43,60,66,86,102,132,173] sprawiły, że dla przeciętnego programisty robota definiowanie ścieżki robota w przestrzeni jest zadaniem złożonym i trudno przyswajalnym. Stopień uwikłania ogólnej postaci modelu matematycznego manipulatora robota niezbędnego do wyznaczania współrzędnych ruchu (model kinematyczny robota) nie pozwala na ich prostą,

zalgorytmizowaną interpretację i bezpośrednie zastosowanie. Obecnie rozwiązanie tych zagadnień, drogą w miarę skutecznych algorytmów komputerowych (metody numeryczne [5,8,41,43,52,102,120,132,152]), mimo wielu trudności, nie stanowi większego problemu. Poważniejszym zadaniem okazała się konieczna w tym przypadku umiejętność programowego (symbolicznego) definiowania warunków geometrycznych pożądanych przemieszczeń robota w zewnętrznej przestrzeni stanu.

Tekstowe języki programowania robotów nie znalazły więc trwałego miejsca w grupie metod komputerowego wspomagania programowania robotów przemysłowych. Takie języki programowania off-line robotów przemysłowych, jak: AL, Pasro, AML, ARM, czy wiele innych [13,14,59,102,115,161,209], jak się wydaje, nie znalazły powszechnego zastosowania przemysłowego [14,102,110]. Ich możliwości ograniczają praktyczne względy. Dlatego też najskuteczniejsza nadal pozostaje metoda programowania robota techniką samouczenia.

Czynione próby z językami programowania robotów pokazały jednak pewną prawidłowość. Mianowicie udowodniły, że:

- sam język w postaci symbolicznej, nawet w najbardziej rozwiniętej postaci semantycznej i syntaktycznej, nie zapewnia osiągnięcia pożądanego sukcesu praktycznego,
- niezbędnym elementem komputerowego wspomagania programowania robotów przemysłowych pozostaje w dalszym ciągu wizualna ocena planowanych w przestrzeni przedsięwzięć robota, pozwalająca na jednoznaczną ocenę warunków jego kolizyjności z obiektami sceny.

Presja wymaganej przez rynek produkcji elastycznej oraz szybki rozwój mikroinformatyki i grafiki komputerowej sprawiły, że w technikach komputerowego wspomagania programowania robotów przemysłowych pojawiły się nowe trendy. Wyznaczają je systemy graficzne CAD-3D, wykorzystujące algorytmy geometrii obliczeniowej [151], w których definiowanie warunków geometrycznych środowiska technologicznego zrobotyzowanego systemu technologicznego (wirtualny model sceny) i samego robota oparte jest na modelowaniu graficznym, typowym dla wysoko rozwiniętych systemów CAD. Systemy te wyposażone w odpowiednie:

- proste języki programowania robotów (np. ML, MELFA-BASIC, SRPL, IRL, MRL [32,47,67,159,160]), korzystające przy definiowaniu ruchu robota z danych uzyskanych drogą graficznego wykreślenia na ekranie programatora punktów kształtujących poszukiwaną ścieżkę w przestrzeni zadaniowej,

- algorytmy kinematyki i dynamiki robota [43,110,138,139,185,186,187], typowe dla zaawansowanych systemów CAD (np. CATIA [77]),
- bazy danych struktur kinematycznych robotów przemysłowych,
- interfejsy umożliwiające transmisję opracowanych programów do układów sterowania robota,

dają możliwość dogodnego zaprogramowania robota do działania w rzeczywistych warunkach produkcyjnych. Dodatkowo umożliwiają:

- w miarę pełną i dokładną wizualizację otoczenia robota i typowanie miejsc w przestrzeni, wyznaczających pożądaną ścieżkę, uwzględniającą konieczność omijania przeszkód (na tyle precyzyjne, na ile precyzyjny jest budowany w nich wirtualny model sceny),
- określenie związków (powiązań geometrycznych) pomiędzy obiektami technologicznymi sceny robota i przedmiotami manipulowanymi, oprzyrządowaniem technologicznym robota i samym robotem,
- synchronizację robota z otoczeniem, przez kreowanie prostych relacji czasowych (kolejnościowych) i informacyjnych (sygnały binarne) pomiędzy obiektami, z robotem,
- definiowanie użytych układów sensorowych,

przez co stwarzają mu w sposób wirtualny dogodne warunki „normalnej" pracy z robotem, a to zwalnia go z konieczności stosowania sformalizowanych złożonych zapisów definiowania planowanego ruchu robota w przestrzeni 3D.

Do grupy tych systemów należy zaliczyć: IGRIP (Advanced Automation Technologies, Inc., USA [67]), COSIMIR Professional i COSIMIR VR (EF-Robotertechnik, Niemcy [32]), Robcad/eMPower i eM-Workplane (Tecnomatix, USA [47]), czy: EGEMIN (Niemcy [212]), Robotworks [212], KUKA Roboter (Niemcy [212]), ALMA [212] oraz Robotmaster [159], czy Robsim (CAMELOT, Dania [160]).

Z doświadczeń zebranych przez autora w pracy z systemem COSIMIR i programistów systemów eMPower (Robcad) oraz IGRIP (LCD Technologies, Szwajcaria) wynika, że mimo tak wielkich możliwości, wśród których jedną z najważniejszych jest możliwość wizualnej oceny planowanych działań robota i ich skutków, systemy te wykazują jednak pewne braki. Należy do nich w dalszym ciągu: planowanie logiki programu oraz trajektorii (wyznaczanie położeń docelowych i pośrednich oraz wskazywanie sposobów obejścia przeszkód, prędkości ruchu robota itd.) przez programistę (stąd wskazana wcześniej konieczność dopracowania gotowego programu off-line robota i dopracowania go w warunkach rzeczywistych sceny).

Dzieje się tak dlatego, że w dalszym ciągu istotnym problemem jest precyzja typowania położeń pośrednich pomiędzy położeniem początku i końca poszukiwanej ścieżki, pozwalających na omijanie przeszkód, co ma szczególne znaczenie w położeniach związanych z obsługą stanowisk, a dotyczących pobierania i odkładania przedmiotu manipulowanego. Jakość i precyzja programowanych ruchów robota i orientacja wynikają z precyzji opracowanego modelu (jego budowa, jak łatwo się domyślić, jest pracochłonna, więc jest on zawsze jakimś – większym lub mniejszym uproszczeniem warunków rzeczywistych) oraz wyobraźni przestrzennej programisty. Problem skali wirtualnego modelu graficznego i jego dwuwymiarowy podgląd na ekranie programatora robota często nie daje dostatecznej pewności co do precyzji geometrycznej konfigurowanej ścieżki robota. Oznacza to, że samo wyznaczanie przestrzennego kształtu planowanej ścieżki robota, pozwalającej na omijanie przeszkód, czy penetrację obszarów przestrzeni o dużym ich zagęszczeniu (np. przestrzeń robocza obrabiarki, stanowiska odkładcze w magazynach, itp.), nie daje gwarancji stuprocentowego powodzenia tych działań. Z tych powodów systemy te jedynie uzupełniają podstawową technikę programowania samouczącego, a wszystkie czynności związane z planowaniem trajektorii i opracowaniem algorytmu pracy robota, wynikającego z obsługiwanego procesu technologicznego, pozostawiają w dalszym ciągu do rozwiązania programiście.

Brak w systemach programowania off-line robotów klasy CAD (COSIMIR Professional, Robcad, IGRIP) rozwiązanych zagadnień planowania akcji i bezkolizyjnych trajektorii jest dość istotny. Istniejąca w nich opcja *„kontrola kolizyjności"* ogranicza się tylko do sprawdzenia – porównania (określonych myszką) współrzędnych ustalonych intuicyjnie położeń robota w wirtualnej przestrzeni sceny i wskazania tych, dla których zachodzi incydencja z obiektami sceny. Sposób ich skorygowania należy już do programisty. Z kolei wyznaczone takim sposobem, stosunkowo „długie" odcinki pomiędzy kolejnymi położeniami pośrednimi planowanej ścieżki nie podlegają już żadnej weryfikacji. To wymusza bezwzględną konieczność sprawdzenia opracowanego off-line programu robota w warunkach rzeczywistych. Oczywiste jest, że sposób wybranego przez programistę obejścia przeszkody jest czysto intuicyjny i nie podlega żadnej ocenie systemu programowania. Stąd też podobieństwo tych systemów do metody programowania samouczącego jest bardzo bliskie.

Wydaje się zatem, że niezbędnym elementem składowym omawianych systemów programowania off-line typu CAD jest wyposażenie ich w dodatkowe możliwości, do których należy zaliczyć:

- ustalanie kolejności zadań robota w systemie zrobotyzowanym, oparte na technikach harmonogramowania (cyklogram),
- automatyczne planowanie ścieżek w złożonej przestrzeni zrobotyzowanego systemu technologicznego (przestrzeni zadaniowej), wyposażone w mechanizm oceny ich kolizyjności, oparte na analizie geometrycznej środowiska robota i wyznaczeniu dla istniejących warunków geometrycznych możliwie najlepszych, czyli optymalnych (najczęściej w sensie lokalnym, tzw. suboptymalnych, np. najkrótszych [40,49,50,56,62,114,140,142,170]) ścieżek, po których przemieszcza się robot w przestrzeni,
- optymalizacja planowania trajektorii robota, oparta na przyjętych jednoznacznie kryteriach (kryteria optymalizowania, w tzw. sensie globalnym, generowanych trajektorii robota nie są jednoznacznie ustalone, choć literatura [15,37,46,48,49,56,58, 61,62,73,74,133,134,136,144,145,156,164,165,213,217,220] określa trzy kryteria optymalizacji: minimum drogi, minimum czasu i tzw. energetyczne, czyli minimum energii potrzebnej na zrealizowanie zaplanowanej ścieżki), co pozwalałoby na zdefiniowanie bezkolizyjnych trajektorii robota.

3. Planowanie zadań robotów

Zadanie programowania robota przemysłowego składa się z dwóch zadań cząstkowych [90]. Pierwszym z nich jest zadanie podstawowe programowania ruchu robota w zewnętrznym środowisku (scenie), drugim zaś jest zadanie pomocnicze, obejmujące proces definiowania zasad współpracy (komunikacji) robota z otoczeniem technologicznym, gwarantujących spełnienie warunku wzajemnej i pełnej synchronizacji [84,135,157]. Zadanie podstawowe programowania robota związane jest z problemem planowania jego trajektorii w przestrzeni sceny.

Ogólnie, *zadanie planowania trajektorii robota* [36,42,43,70,73,110,111,126,133,170] (w skrócie nazywane *zadaniem planowania*, a omawiane metody - *metodami planowania*) określane jest, w najprostszej postaci, jako definiowanie sparametryzowanej czasem funkcji przejścia robota z położenia początkowego, pomiędzy wskazanymi położeniami pośrednimi, do położenia końcowego, określonego w przestrzeni konfiguracyjnej robota [4,43,69,102,110,126,193,215]. Zadanie to powinno być rozwiązywane z uwzględnieniem ograniczeń wynikających z [114,217]:

- wielkości i kształtu przestrzeni zajętej przez przeszkody robota, czyli przestrzeni kolizyjności robota,
- kinematyki i dynamiki łańcucha kinematycznego robota [43,102,138,139,177,186, 187,200,201],
- algorytmu sterowania robotem,
- wymaganego kształtu trajektorii robota.

Czynniki te wyznaczają również kryteria optymalizacyjne zadania planowania [144].

W zależności od obszaru, w którym proces planowania przebiega, przestrzenią konfiguracyjną może być zewnętrzna przestrzeń zadania, lub wewnętrzna przestrzeń stanu robota związana z procesem sterowania robotem. Z przytoczonej definicji zadania planowania wynika, że problem planowania trajektorii robota sprowadza się do wyznaczenia pewnego

równania matematycznego wzorca drogi (tzw. funkcji przejścia [42,43,69,110]), z czasem jako parametrem. Równanie to, określone w wewnętrznej uogólnionej przestrzeni konfiguracyjnej robota, pozwala na wyznaczenie parametrów sterujących robotem względem czasu, co powinno gwarantować zachowanie wszystkich istotnych kryteriów geometrycznych i dynamicznych wyznaczonego przemieszczenia [43,102,110,186,187]. Dla zadania planowania, rozwiązywanego w zewnętrznej przestrzeni zadania, definicja ta wskazuje, że poszukiwana funkcja przejścia rozpięta jest pomiędzy wskazanymi położeniami pośrednimi (tzw. punktami węzłowymi P_{wi} [43,110]), leżącymi pomiędzy położeniem startowym a końcowym planowanego ruchu. Punkty te określają krzywą (łamaną), zwaną *ścieżką robota*, po której przemieszcza się tzw. *punkt charakterystyczny pozycjonowania robota* [69,110,132] związany z ostatnim elementem ostatniego ogniwa łańcucha kinematycznego manipulatora robota. Ścieżka odniesiona do wewnętrznej uogólnionej przestrzeni stanów wewnętrznych robota nosi miano *toru* [43]. Poszukiwanie punktów węzłowych P_{wi} jest zatem początkowym etapem procesu planowania i sprowadza się do wyznaczenia takiego uporządkowanego zbioru danych geometrycznych, definiujących te punkty w przestrzeni programowania, który by gwarantował ich osiągnięcie przez robota, tym samym spełniałby ograniczenia kinematyczne mechanizmu robota. Zbiór wszystkich punktów węzłowych pomiędzy położeniem startu a końca planowanej trajektorii wyznacza tzw. rozwiązanie dopuszczalne (możliwe do zrealizowania) poszukiwanej ścieżki robota w przestrzeni zadania, które stanowi zbiór danych wejściowych do budowy funkcji przejścia.

Ścieżka ta, ze względu na konieczność ominięcia napotykanych przeszkód, najczęściej jest łamaną, której poszczególne odcinki ograniczone są kolejno wyznaczonymi punktami węzłowymi. Oznacza to, że problem planowania bezkolizyjnych trajektorii robota rozpoczyna się od rozwiązania zadania wyznaczenia bezkolizyjnej ścieżki w przestrzeni określonej zakresem sceny [197]. Z postaci geometrycznej wyznaczonej ścieżki (łamana) wynika, że ruch robota odbywa się etapami, które określają kolejne punkty węzłowe. Można powiedzieć, że odcinki te (wyznaczające ścieżkę robota) określają kolejne kroki t_i, jakie robot będzie realizował przechodząc z położenia początkowego do końcowego. Ich suma określa wyznaczoną łamaną, czyli znalezioną bezkolizyjną ścieżkę robota:

$$\Gamma = \bigcup_{i=1}^{k} t_i$$

gdzie: k – liczba wszystkich odcinków wyznaczonej ścieżki robota (kroków).

W metodach zajmujących się tym zagadnieniem najtrudniejsze jest rozwiązanie problemu analizy orientowania się robota w otaczającej go przestrzeni, charakteryzującej się dużą złożonością geometryczną. Potęguje się on znacznie w przypadku podjęcia prób jego algorytmizacji, co ma miejsce w komputerowych systemach programowania off-line robotów.

Zewnętrzna przestrzeń zadania (a z nią przestrzeń programowania) opisana jest w kartezjańskim układzie współrzędnych [43,69,110,165], przez co rozwiązanie zadania planowania bezkolizyjnych ścieżek robota prowadzi do wygenerowania zbioru danych geometrycznych określających punkty węzłowe poszukiwanej bezkolizyjnej ścieżki.

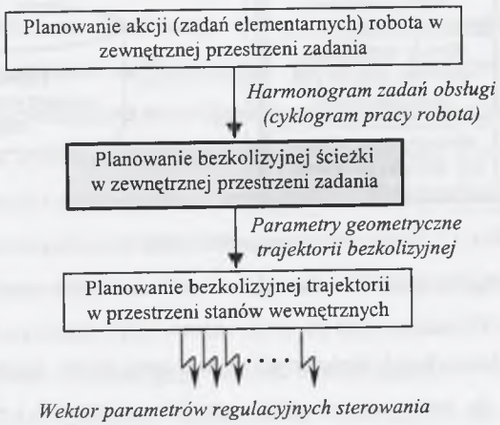
Z kolei rozwiązanie zadania planowania w wewnętrznej przestrzeni stanów jest związane z procesem sterowania robotem i tzw. wektorem współrzędnych konfiguracyjnych (wektor współrzędnych naturalnych – uogólnionych robota) [43,102,132], określających przemieszczenie względne poszczególnych ogniw manipulatora robota, związane z każdą sterowaną osią jego łańcucha kinematycznego, a przez to zbiór parametrów sterujących napędami robota. Przestrzeń ta opisana jest układem wewnętrznych współrzędnych konfiguracyjnych, których liczba określona jest ruchliwością łańcucha kinematycznego robota [79,87,131,200,201].

W robotyce istnieją metody planowania bezkolizyjnych trajektorii robotów przeznaczone dla robotów stacjonarnych i mobilnych oraz rozwiązujące problem planowania zarówno w przestrzeni stanów zewnętrznych robota, jak i wewnętrznych [43,69,102,110,171].

W odniesieniu do stacjonarnych robotów manipulacyjnych istniejące i najczęściej spotykane są geometryczne (grafowe) [4,9,15,35,43,48,57,69,75,96,106,110,114,123,134, 168,193,197,203,204,205] metody planowania trajektorii. Oparte są one na analizie geometrycznej otoczenia robota i rozwiązują ten problem techniką bazującą na uporządkowanym podziale (aproksymacji) analizowanej przestrzeni określonymi obiektami geometrycznymi, tzw. podprzestrzeniami elementarnymi [110,132].

Należy zaznaczyć, że odmiennym jakościowo problemem planowania działań robota jest tzw. problem planowania akcji robota [69,110,115,169,171]. Zagadnienie to koncentruje swoją uwagę na automatycznym (komputerowym) wyznaczeniu i konfigurowaniu poprawnej sekwencji czynności elementarnych robota. Czynności elementarne, rozumiane jako podstawowe ruchy robota, identyfikowane z jedną instrukcją programową robota, realizowane w zewnętrznej przestrzeni zadania, powinny umożliwić obsługę stanowisk technologicznych zgodnie z wymogami procesu technologicznego. Nietrudno zauważyć, że problem ten ściśle wiąże się z zadaniem harmonogramowania procesu obsługi stanowisk sceny realizowanych przez robota, przez co stanowi punkt wyjścia do określenia algorytmu (harmonogramu) pracy

i algorytmu programu robota. Zagadnienie to w literaturze robotycznej reprezentowane jest jako tzw. problem układania klocków [59,156,211] i rozwiązywane metodami sztucznej inteligencji. Algorytm programu robota wskazuje kolejno sekwencje miejsc i zadań (tzw. cyklogram pracy robota), które musi on osiągnąć i wykonać, chcąc zrealizować zaplanowane procesem technologicznym czynności obsługi. Na tej podstawie, w oparciu o metody planowania trajektorii, może być wyznaczona bezkolizyjna ścieżka robota. Ta zaś, jak już wspomniano, powinna pozwolić na otrzymanie sparametryzowanej czasem funkcji przejścia oraz jej optymalizację wg przyjętych kryteriów, tym samym zdefiniować poszukiwaną trajektorię robota i w konsekwencji wyznaczyć wektor parametrów sterujących napędami robota. Proces ten pokazano na rysunku 3.1.



Rys. 3.1. Hierarchia zadań planowania akcji i trajektorii robota [69,126]

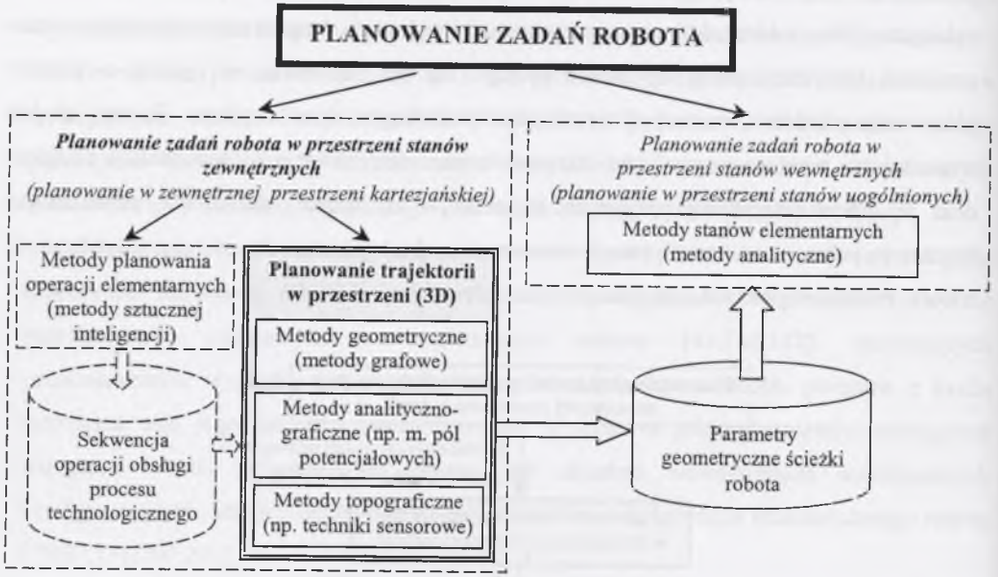
Fig. 3.1. Priority of the tasks involving the planning of robot operations and trajectory [69, 126]

W dalszej części pracy pominięte zostaną metody planowania akcji robota, a uwagę skoncentrujemy na zagadnieniu zadaniu planowania bezkolizyjnych ścieżek robota w zewnętrznej przestrzeni zadania.

3.1. METODY PLANOWANIA TRAJEKTORII ROBOTÓW

Opisywane w literaturze zagadnienie planowania trajektorii robota [35,43,110,111,197], w zależności od jego typu, sposobu zbierania i definiowania danych o planowanej drodze i jego otoczeniu oraz szczegółowości i zakresu proponowanych rozwiązań, można ogólnie

podzielić na kilka grup, których obszar zastosowania jest ściśle określony. Każde z nich dzielą się z kolei na bardziej szczegółowe techniki planowania, co ilustruje rys. 3.2.



Rys. 3.2. Metody planowania zadań robota

Fig. 3.2. Methods of planning tasks performed by the robot

Ze względu na różnorodność istniejących metod oraz temat zainteresowania niniejszej pracy ograniczono się do metod związanych z analizą przestrzeni i planowania trajektorii przeznaczonych dla robotów przemysłowych stacjonarnych i manipulacyjnych.

W ogólnym przypadku metody te, wykorzystując i przetwarzając informacje o otoczeniu robota, umożliwiają wydzielenie w niej obszarów określających tzw. przestrzeń bezkolizyjną robota, w której możliwe jest wyznaczenie ścieżki bezkolizyjnej poprzez określenie współrzędnych punktów węzłowych pomiędzy jego położeniem bieżącym a wskazanym przez programistę jej położeniem docelowym. Metody te koncentrują się na rozwiązaniu tego zadania w kategoriach ruchu zgrubnego i dokładnego (planowanie globalne i lokalne) [38,76,106,135,168,197], tzn. bez uszczegółowiania proponowanych rozwiązań w obszarze orientacji kiści robota, co związane jest z ruchem precyzyjnym robota [135]. Dzieje się tak dlatego, że de facto problem kolizji robota z otoczeniem powinien być rozpatrywany w kontekście całego jego łańcucha kinematycznego. Jednak, ze względu na konieczne w tym celu analizy, w metodach planowania bezkolizyjnych trajektorii robotów, łańcuch kinematyczny manipulatora robota redukuje się do tzw. punktu charakterystycznego

pozycjonowania [93,98,110,132] związanego z jego kiścią (efektorem). Tym samym ograniczenie to pozwala wyeliminować z zadania planowania problem orientacji kiści w wyznaczonych metodą położeniach. O ile położenie końca poszukiwanej ścieżki zadane jest a priori warunkami procesu technologicznego (sekwencją realizowanych zadań), o tyle otwarty pozostaje problem sposobu, w jaki to położenie końcowe może zostać osiągnięte (w sensie dojścia do celu). Odpowiedź na to pytanie stanowi właśnie istotę zadania planowania w zewnętrznej przestrzeni zadaniowej robota i sprowadza się do rozwiązania następujących zagadnień [110,114,127,144]:

- analizy przestrzeni, czyli rozpoznania przestrzeni wokół robota i wydzielenia w niej obszarów bezkolizyjnych, w których może być realizowany potencjalny ruch robota,
- wyznaczenia wg określonej strategii działania (np. sposobu omijania przeszkód), dla zadanego położenia docelowego i startowego (bieżącego) zbioru potencjalnie możliwych rozwiązań dopuszczalnych [49,110], określonych sekwencją położeń pośrednich, pozwalających na omijanie przeszkód (zbiór punktów węzłowych),
- zdefiniowania kryterium optymalizacyjnego planowanej ścieżki (lokalnego, zwanego również prostym) pozwalającego na wybór ze zbioru rozwiązań dopuszczalnych tej najlepszej (rozwiązanie suboptymalne).

Potencjalnie duża różnorodność możliwych do wykorzystania ścieżek omijania przeszkód sprawia, że zadanie planowania należy do klasy zadań NP-trudnych [43,169,171] i wymaga wykorzystania specjalnych technik pozwalających na wybór rozwiązania spełniającego założone kryterium optymalności, należących do metod przeszukiwania heurystycznego. W warunkach programowania samouczącego oraz systemów robotowych klasy CAD zadanie to rozwiązywane jest intuicyjnie, w oparciu o doświadczenie programisty i polega na arbitralnym wyborze rozwiązania intuicyjnie uznanego za najlepsze, jeżeli tylko jest ono możliwe do zrealizowania z punktu widzenia kinematyki robota. Jedynym wyraźnym kryterium pozostaje więc poprawność realizowanych ścieżek, rozumiana wyłącznie w kategoriach unikania kolizji i wyboru tej ocenianej jako najkrótsza bądź najprostsza. To zaś oznacza najczęściej wybór ścieżki najłatwiej osiągalnej z położenia startowego.

Jak pokazano na rys. 3.2, metody planowania dzielą się na takie, które rozwiązują zadanie planowania na płaszczyźnie (2D) i w przestrzeni (3D):

- techniki 2D [9,15,37,43,57,60,106,111,136,158,190,213] koncentrują się na problemach związanych z robotami mobilnymi, dla których przestrzeń ruchu opisują dwie współrzędne (najczęściej kartezjańskie lub dowolne, transformowane do układu

kartezjańskiego) oraz warunkowo jeden kąt obrotu wokół osi prostopadłej do płaszczyzny, po której porusza się robot (płaszczyzna podstawy robota),

- techniki 3D [12,43,48,77,110,134,163,172,173,203,204,205,215] dotyczą głównie manipulacyjnych robotów stacjonarnych, dla których przestrzeń działania jest konstrukcyjnie związana z przestrzenią 3D (przestrzenne łańcuchy kinematyczne) i dla których współrzędne orientacji ruchu robota (orientacji punktu charakterystycznego pozycjonowania związanego z ostatnim elementem ostatniego ogniwa łańcucha kinematycznego robota) są pomijane.

3.2. ANALIZA PRZESTRZENI

Podstawowym zadaniem w obrębie metod planowania (niezależnie od metod 2D lub 3D) jest zadanie analizy przestrzeni wokół robota, związanej ze sceną, w której się on porusza. Analiza ta jest jednym z najtrudniejszych problemów, jakie należy rozwiązać, stwarzając możliwość automatycznego generowania (wyznaczania) zbioru położeń pośrednich ścieżki robota w przestrzeni (lub na płaszczyźnie). W ogólnym przypadku zadanie to obejmuje grupę następujących zadań cząstkowych:

- sposobu definiowania i analizy obiektów technologicznych, z którymi robot współpracuje i które są jego potencjalnymi przeszkodami,
- analizy danych geometrycznych modelu świata robota dla potrzeb zadania planowania,
- określenia warunków oceny kolizyjności robota z otoczeniem i strategii wyznaczania ścieżek obejściowych.

Zadania z tej grupy łączą się w jedną całość, dając możliwość uzyskania danych geometrycznych, na bazie których można wytypować dostępne i możliwe do zrealizowania ścieżki, pozwalające na wyznaczenie przejścia z położenia startowego (bieżącego) do wskazanego położenia końcowego. Zależnie od technik, jakie się stosuje do rozwiązania problemu analizy otoczenia robota, istniejące metody dzielą się na [43,69,110,132]:

- geometryczne, wykorzystujące zapisany odpowiednimi procedurami formalnymi (język) lub graficznymi (zaawansowane systemy CAD) geometryczny model świata robota (model wirtualny),
- topograficzne, pozwalające na budowanie aktualnych map topograficznych terenu (obszaru), po którym porusza się robot, w oparciu o złożony system sensorowy.

3.2.1. Metody topograficzne

Metody topograficzne [15,36,38,61,76,77,119,205] oparte są na mechanizmie budowania mapy terenu (mapy dróg), w którym porusza się robot. Wykorzystują do tego celu odpowiednio zaawansowane:

- techniki i układy sensorowe, takie jak: kamery TV (mono- lub lepiej stereowizyjne), techniki pomiaru odległości lub częściej układy sensorowe oparte na kamerze TV + układ sensorów odległościowych [31,44,81,82,103,109,119,135,154,194], pozwalające na penetrację informacyjną przestrzeni wokół robota,
- techniki pozyskiwania informacji geometrycznej o otoczeniu robota pozwalające na globalną analizę i interpretację jego otoczenia (pełny model geometryczny sceny).

Ze względu na bardzo złożone i niejednoznacznie określone układy wykorzystywanych danych (wzajemnie niekompatybilne) o otoczeniu, w którym:

- planowany jest ruch (współrzędne kartezjańskie),
- zbierana jest informacja o otoczeniu przez różnego typu układy sensorowe (najczęściej: 1D lub 2D),

metody te praktycznie nie są stosowane w zrobotyzowanych systemach produkcyjnych do wspomagania sterowania ruchem robotów manipulacyjnych. Wspomniany problem wzajemnej niekompatybilności danych sensorowych o ruchu robota wynika z tego, że nie mają one wspólnego (wzajemnie kompatybilnego) jednoznacznie określonego układu odniesienia. Nof [135] wyróżnia w procesie analizy danych o ruchu robota przestrzeń współrzędnych programowania i sensorowy 1D, 2D i rzadko 3D, wskazując, że główną przyczyną trudności ich wykorzystania jest brak ich wzajemnej relacji. Tym samym niezbędne jest stosowanie, często złożonych obliczeniowo, procedur wzajemnej transformacji tych danych, tak by mogły być wykorzystane do modyfikowania ruchu robota. Spostrzeżenia te potwierdzają również autorzy prac [31,44,81,82,106,153,194]. Każdy wykorzystany do tego celu typ układu sensorowego wyznacza więc inny układ odniesienia i określa innego typu informacje, które powinny być wykorzystywane przy typowaniu przestrzeni bezkolizyjnej w zadaniu planowania ścieżki robota.

Informacje o otoczeniu, pozyskiwane z układów sensorowych, wymagają więc uzupełnienia ich o relacje wzajemnego odniesienia, porównania i definiowania jednoznacznych związków z aktualnym położeniem robota względem potencjalnych przeszkód [135]. Mało precyzyjne i stosunkowo powolne algorytmy analizy tej informacji (np. algorytmy analizy obrazu [31,44,81,82,117,153,189]) oraz niejednoznaczność

wzajemnych relacji zbieranych danych sensorowych sprawiają, że metody te są jak na razie mało skuteczne.

Jedynym obszarem szerszego zastosowania złożonych układów sensorowych w robotyce są roboty mobilne (wymagające informacji „tylko" z przestrzeni R^2), gdzie problem orientacji ogranicza się do jednego kierunku ruchu, przez co oddziaływanie układów sensorowych ogranicza się do pomiaru odległości od przeszkód i analizy obszaru widzenia w ściśle wyznaczonych kierunkach. Układ wizji maszynowej wymaga, oprócz odpowiednio sprawnych (dokładnych i jednoznacznych) algorytmów analizy uzyskiwanych obrazów, również odpowiedniego sposobu przełożenia tej informacji na parametry geometryczne postrzeganego nim otoczenia. To sprawia, że są one praktycznie bardzo rzadko stosowane w robotach stacjonarnych, a oprócz stosunkowo drogiego układu wizyjnego wymagają do analizy napływających danych zastosowania odpowiednio zaawansowanego oprogramowania, rozbudowanych baz danych (tzw. baz wzorców geometrycznych otoczenia) i współpracy układu sterowania robota z nadrzędnym systemem komputerowym, dokonującym niezbędnych analiz zbieranych danych. Problem ten komplikuje w znacznym stopniu dynamika zmian w otoczeniu robota i potencjalne zakłócenia, które znacząco utrudniają zbieranie i filtrowanie otrzymywanych danych i bardzo często wymagają spełnienia dodatkowych, rygorystycznych warunków organizacyjnych (np. odpowiednie oświetlenie, kilka kamer, wzajemnie przełączalnych, co umożliwia analizę obrazów tylko w wybranych miejscach i pozwala uprościć procedury analizy danych, itd.) [135,189,194].

Rozbicie układu sensorowego, analizującego przestrzeń na grupy (najczęściej) jednorodnych sensorów penetrujących wybrane obszary przestrzeni, jest obecnie najlepszym i najszybszym sposobem na wykorzystanie pochodzącej z nich informacji w trybie programowania ruchu on-line. Dzięki takiemu podejściu uzyskuje się [31,44,135,154]:

- jednorodność zbieranych informacji związanych z lokalnie występującym problemem,
- znaczne ułatwienie analizy danych poprzez zawężenie obszaru możliwych rozwiązań do jednego problemy.

Przykładem może tu być wykorzystanie np. kamery TV do wykrywania i analizowania ułożenia dowolnie podawanych przedmiotów w wybranym miejscu na podajniku taśmowym. Dzięki takiemu rozwiązaniu problemu możliwe jest stosunkowo szybkie skorygowanie ruchu robota przez dopasowanie orientacji kiści do przypadkowego układu przedmiotu.

Jednakże ogólnie, ze względu na powolność analizy zbieranych danych, metody topograficzne wymuszają zastosowanie odpowiednich algorytmów działania systemów planowania [86,113,135]. Polegają one na wykorzystaniu stosownego systemu:

a) po pierwsze: sterowania ruchem robota, którego elementami są odpowiednie procedury aktywizujące wybrane sensory rozlokowane w otoczeniu robota, przez co napływająca informacja jest precyzyjnie związana z określonymi, newralgicznymi miejscami w przestrzeni. Mechanizm ten znacznie upraszcza proces zbierania danych przez ich ograniczenie, jak również przez sekwencyjne wybieranie kolejno odpowiednich układów analizy wybranych obszarów przestrzeni. Sprawia to, że napływająca informacja jest bardziej jednorodna, co ułatwia jej analizę i wyciąganie właściwych wniosków odnośnie do zachowania robota,
b) po drugie: programowania robota pozwalającego, ze względu na powolność analizy danych, na wygenerowanie 2-3 instrukcji robota [86,135], w czasie realizacji których układ sterowania nadrzędnego ma możliwość zebrania, przeanalizowania i zdefiniowania danych, pozwalających na wyznaczenie 2-3 kolejnych instrukcji ruchu robota.

Mimo tych dość znaczących wad, metody te są bodajże jedynymi, które mogą być zastosowane w dynamicznym środowisku, a tym samym jedynie one pozwalają na wykorzystanie ich w systemach programowania robotów on-line czasu rzeczywistego, co jest związane obecnie głównie z robotami mobilnymi.

W tych warunkach najwięcej nadziei, w stosunku do planowania ruchu robotów stacjonarnych i manipulacyjnych, wiąże się z metodami geometrycznymi. Są one technicznie znacznie prostsze.

3.2.2. Metody geometryczne

Podstawową zaletą metod geometrycznych jest jednoznaczność danych, niezbędnych do przeprowadzenia stosownych analiz geometrii przestrzennej. Metody te, w oparciu o istniejący model świata robota, czyli geometryczny opis otoczenia robota, wymagają odpowiedniej:

- techniki, pozwalającej na analizę otoczenia robota,
- strategii, umożliwiającej wyznaczenie zbioru rozwiązań dopuszczalnych w analizowanej przestrzeni,
- techniki przeszukiwania i analizy uzyskanego zbioru rozwiązań dopuszczalnych (danych), w celu znalezienia rozwiązania optymalnego (najczęściej suboptymalnego), zgodnie z przyjętym kryterium optymalności (najczęściej proste kryterium najkrótszej drogi).

Wadą tych metod jest konieczność opracowania strategii, według której poszukiwana jest ścieżka robota w przestrzeni w zbiorze danych geometrycznych, określających zdefiniowany symbolicznie model świata robota. Strategia poszukiwania potencjalnych rozwiązań dopuszczalnych związana jest ściśle z techniką opisu wykorzystanej przestrzeni. Opiera się na założeniu, że każde potencjalnie możliwe do uzyskania przejście (intuicyjnie nawet najmniej racjonalne) pomiędzy położeniem startowym a końcowym planowanej ścieżki jest dopuszczalne. To zaś wymaga zastosowania odpowiednio skutecznej metody wyboru rozwiązania najlepszego, drogą selekcji i eliminacji rozwiązań złych, lub nie dość dobrych, w świetle przyjętych kryteriów optymalnościowych poszukiwanej ścieżki. W istniejących metodach opartych na przybliżonym podziale przestrzeni obejmującej scenę robota (metody rastrowe), czyli na jej aproksymacji, elementami opisującymi ją są typowe obiekty geometrii przestrzennej, takie jak [29,35,43,48,110,134,164,203,204]: sześciany, kule, elipsoidy czy wielościany wypukłe (np. trapezoidy). Tym samym zakłada się konieczność dokonania oceny przydatności każdego powstałego w wyniku analizy rastra [12,43,110] z punktu widzenia jego zajętości przeszkodami robota. Spośród tych, które nie są zajęte przez przeszkody, wybiera się rastry, mogące wskazywać punkty węzłowe poszukiwanej bezkolizyjnej ścieżki robota. Wymaganą warunkami sceny i zadania konieczną dokładność planowanego ruchu robota osiąga się drogą odpowiednio daleko idącego uszczegółowienia koniecznej analizy geometrycznej (metody te zaliczane są do grupy metod przybliżonych [43,110]). Stan ten osiąga się przeprowadzając dalsze podziały istniejących już rastrów. Układ wzajemnie związanych rastrów zawartych w analizowanej przestrzeni sceny tworzy *przestrzeń poszukiwań* [43,110]. Takie podejście powoduje znaczącą rozbudowę przestrzeni poszukiwania rozwiązania optymalnego, czyli powstałego zbioru rozwiązań dopuszczalnych, który tworzy tzw. graf - *drzewo decyzyjne poszukiwań* [16,43] (w przypadkach bardzo złożonych czasami las drzew). Jego korzeń stanowi pierwotny atom określający całą przestrzeń robota poddaną analizie, a wierzchołki rodzicielskie i liście stanowią atomy powstałe kolejno na niższych (wcześniejszych) i wyższych (późniejszych) poziomach podziału [12,43,75,104,110,151,163,193,204,205]. Wynikiem kolejnych podziałów jest rozrastanie się przestrzeni poszukiwań, co wynika z koniecznej dokładności procesu planowania. Jego efektem jest znaczące wydłużenie, a tym samym spowolnienie całego procesu analizy i planowania. To zaś wymusza zastosowanie odpowiednich technik przeszukiwania powstałego drzewa decyzyjnego. Wymóg skutecznego przeszukiwania spełniają algorytmy przeszukiwania heurystycznego [4,16,71,72,129,156,180,193,213], w których do zadania planowania wykorzystuje się funkcje heurystyczne zbudowane w oparciu

o przyjęte w zadaniu proste kryterium optymalności, jakim jest najkrótsza ścieżka prowadząca do celu (wyrażana najczęściej miarą euklidesową). Spośród znanych algorytmów heurystycznych w robotyce, zastosowanie znalazły głównie algorytmy przeszukiwania heurystycznego: wszerz (*BFS – Breadth-First-Search*), *BACKTRACKING*, A^*, czy *hill-climbing*. Z tych powodów metody geometryczne zaliczane są również do grupy metod heurystycznych [43], jak również do grupy metod grafowych, ze względu na graf-drzewo decyzyjne, które jest reprezentantem analizowanej przestrzeni.

Metody te charakteryzują się brakiem pamięci działania, co oznacza, że poszukiwanie każdej kolejnej ścieżki rozpoczyna się od najniższego, a kończy na maksymalnie dopuszczalnym, szacowanym i wymaganym dokładnością zadania poziomie podziału.

W wyniku przeszukiwania drzewa decyzyjnego uzyskuje się zbiór punktów węzłowych, na bazie którego konstruuje się funkcję przejścia. Sparametryzowanie jej czasem pozwala w kolejnym etapie planowania uzyskać równanie poszukiwanej bezkolizyjnej trajektorii. Wynika z tego, że metody geometryczne, ograniczając swoje działanie wyłącznie do analizy przestrzeni, są pierwszym etapem planowania.

Z tych też powodów grupę metod „czysto" geometrycznych uzupełniają metody geometryczno-analityczne, w których znaczącym elementem są różne mechanizmy (strategie) wspomagające i upraszczające prowadzoną analizę geometryczną. Ich celem jest określenie minimalnej odległości od przeszkody, gwarantującej bezpieczne przemieszczanie się robota. Do grupy tych metod można zaliczyć metodę sztucznego potencjału [38,43,132,220], czy funkcji kary [43,132]. Metody geometryczno-analityczne, ze względu na bardziej zaawansowany aparat matematyczny, pozwalają na zdefiniowanie funkcji opisującej trajektorię robota, przez co bliższe są zadaniu planowania rozwiązywanego w przestrzeni stanów wewnętrznych.

3.2.2.1. Metody heurystyczne

Grupę metod heurystycznych tworzą wszystkie te metody, które do analizy uzyskanych danych (grafów, drzew) wykorzystują algorytmy przeszukiwania heurystycznego, co wymuszane jest brakiem jednoznacznie określonego wektora poszukiwań drzewa reprezentującego analizowaną przestrzeń, a przez to wielkością zbioru rozwiązań dopuszczalnych. Obejmują one metody geometryczne i analityczne, a tym samym pozwalają na dość skuteczne rozwiązanie zadania planowania zarówno w ujęciu zadania, jak i funkcji stanu [43,69,110,132]. Podstawową cechą metod heurystycznych jest to, że wszystkie one zakładają istnienie otoczenia statycznego robota. Wspomniana (pkt 2.1.3) psudostatyczność

otoczenia robota wynika z tego, że zmiany zachodzą w nim powoli, w wybranych miejscach i są trwałe w dłuższym horyzoncie czasowym. Dlatego, wykorzystując tę właściwość sceny robota, przyjmuje się, że w momencie budowania drzewa decyzyjnego reprezentującego scenę (w momencie rastrowania) jest ona statyczna, przez co dokonywana analiza heurystyczna może zachodzić w warunkach ustalonych, co jest jej podstawowym warunkiem [43,132].

Idea omawianych metod sprowadza się do opracowania i zrealizowania następujących etapów analizy heurystycznej:

- hipoteza: pozwala na wskazanie hipotetycznie najlepszej (spełniającej przyjęte kryterium optymalizacyjne) zaplanowanej ścieżki;
- weryfikacja etap 1°, czyli sprawdzenie poprawności (braku kolizji i wyboru optymalnego) proponowanego rozwiązania: na tym etapie wykorzystuje się funkcję kosztu heurystycznego [16,43,62,126,132,156,205] realizowania wyznaczonego przemieszczenia zdefiniowaną, tak by był on możliwie najniższy. Funkcja kosztów określa kryterium optymalizacji, w którym kosztem może być: minimalny czas przejścia, najkrótsza ścieżka lub jej gładkość [45,46,56,75,124, 132,155];

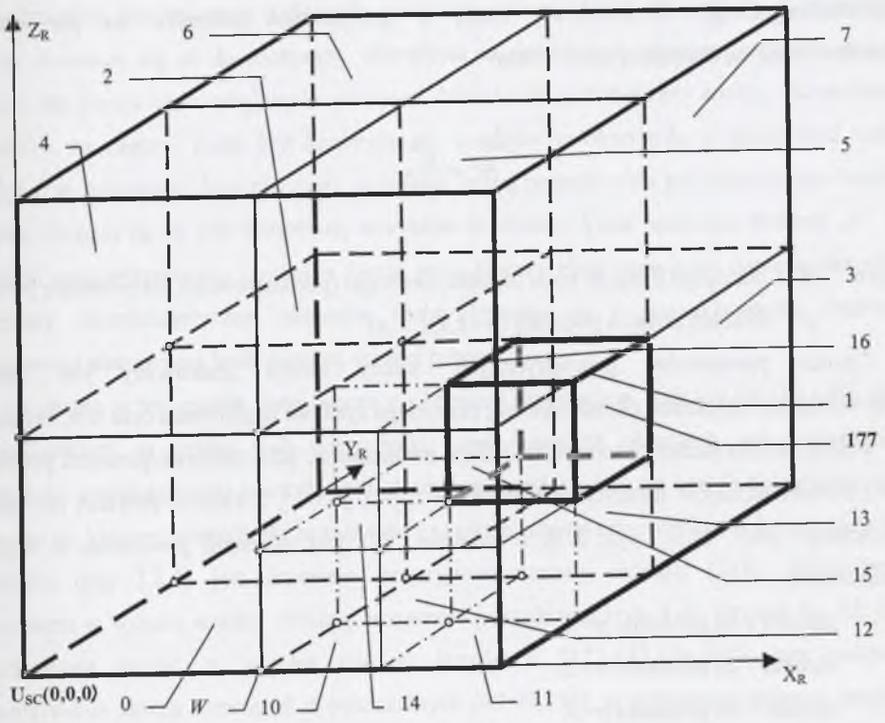
  Na tym etapie wykorzystuje się algorytm przeszukiwania heurystycznego *A** lub jego udoskonaloną wersję *hill-climbing*. Algorytmy te nie gwarantują jednak osiągnięcia zadanego celu planowanej ścieżki w sensie globalnym (dla całej przestrzeni poszukiwań), co jest ich istotną wadą,
- weryfikacja etap 2° - etap ten wprowadza się w celu sprawdzenia warunku, że dwa poprzednie etapy doprowadziły do osiągnięcia zadanego celu (algorytmy *A** i *hill-climbing* nie gwarantują tego). Weryfikacja na etapie 2° jest wolniejsza niż na etapie 1°, ale ponieważ realizowana jest na już wyznaczonej i optymalnej ścieżce, zatem jej działanie jest najkrótsze z możliwych.

3.2.2.1.1. Metoda drzew oktalnych

Metoda ta zasługuje na szczególną uwagę, ze względu na swoje proste założenia, dzięki czemu znalazła zastosowanie zarówno w robotyce [43,118,,163,172,174,204], jak i w grafice komputerowej [151] oraz do analizy przestrzeni 3D [104,203]. Wykorzystuje omówiony już (pkt 3.2.2) mechanizm rastrowania przestrzeni.

Istota metody oktalnej (metoda drzew oktalnych), należącej do grupy metod grafowych (rastrowych) geometrycznych przybliżonych i heurystycznych, opiera się na regularnym podziale przestrzeni na sześciany. Punktem wyjścia jest podstawowy sześcian, w którym

zawarta jest cała przestrzeń sceny [69,110,132,172,174]. Jako punkt wyjścia przyjmuje się przestrzeń obejmującą cały zrobotyzowany system technologiczny. Sześcian ten, zwany sześcianem podstawowym lub początkowym [69,110,132,174], podlega podziałowi, każdorazowo na kolejnych osiem sześcianów (zwanych oktantami), o boku równym połowie boku sześcianu podstawowego (rys. 3.3). Zakłada się, że oktant ten jest „umocowany" w podstawowym kartezjańskim układzie współrzędnych U_{SC} w położeniu *(0,0,0)*, opisującym



Rys. 3.3. Przestrzeń oktalna. *W* – wierzchołek oktantu; cyframi oznaczono indeksy oktantów. Dla większej czytelności rysunku zaznaczono wierzchołki oktantów (kółeczka), a nie ich środki geometryczne

Fig. 3.3. Octal space. *W* – octant vertex, octant indices are designated by digits. For better legibility, the vertices of the octant have been marked (circles) instead of its geometrical centers

całą przestrzeń systemu zrobotyzowanego w taki sposób, że jedno z jego naroży zaczepione jest w początku tego układu, a wychodzące z niego krawędzie pokrywają się z osiami układu U_{SC}. Każdy z dokonywanych podziałów określa tzw. poziom podziału oktalnego, gdzie sześcian podstawowy jest oktantem poziomu „zerowego" $p=0$ (oktant zerowy), a każdy następny wyznacza kolejne poziomy podziałów $p=p+1$ (oktanty poziomu p, tzw. p oktanty).

Każdy z oktantów, w przestrzeni (w układzie sceny U_{SC}), identyfikowany jest z pewnym swoim charakterystycznym elementem, którym może być jeden z jego wierzchołków W lub środek geometryczny. Każdy oktant jest jednoznacznie identyfikowany z tym elementem. Regularność podziałów oktalnych, dokonywanych w przyjętym i niezmiennym układzie współrzędnych U_{SC}, powoduje, że wszystkie parametry geometryczne powstających oktantów podlegają prostym zależnościom rekurencyjnym. Dotyczy to: długości boków oktantów b, ich liczebności na p-tym poziomie podziału i współrzędnych ustalonego elementu charakterystycznego. Przykładowo, każdy z utworzonych oktantów na poziomie p charakteryzuje się długością boku równą:

$$b^p = \frac{b^0}{2^p} \tag{3.1}$$

gdzie: b^0 – określona długość boku oktantu zerowego (podstawowego, na poziomie $p=0$),
p – bieżący poziom podziału ($p=1,2,3,...,n$).

Oprócz parametrów geometrycznych, każdy oktant znakowany jest swoim indywidualnym indeksem, określonym wg przyjętego systemu znakowania oraz tzw. etykietą.

Indeks określa numer oktantu i umożliwia związanie ze sobą zarówno poziomu podziału, jak i numeru oktantów niższych poziomów ($p-1$, $p-2$, $p-3$,...), z których powstał; np. oktant oznaczony jako O_{177} (rys. 3.3), o indeksie *177* jest oktantem powstałym w wyniku podziałów z:

- oktantu *1* na poziomie $p=1$,
- oktantu *7* na poziomie $n=2$,
- oktantu *7* na poziomie $p=3$.

Dzięki takiemu indeksowaniu każdego oktantu można za pomocą wspomnianych zależności rekurencyjnych również określić współrzędne jego położenia w przestrzeni (określonych dla przyjętego elementu charakterystycznego), zależnie od poziomu p podziału oktalnego, na którym powstał (ilość cyfr w indeksie równa jest n, czyli krotności podziałów).

Z kolei etykieta oktantu jest związana z jego zajętością [12,69,132,149,172,174,204]. Omawiana zasada atomizacji trójwymiarowej przestrzeni wokół robota wymuszana jest określeniem stopnia zajętości każdego z powstających w wyniku kolejnych podziałów oktantów. I tak, wykorzystując współrzędne elementu charakterystycznego i długości jego boków oraz porównując te wartości ze współrzędnymi istniejących w przestrzeni obiektów

technologicznych uzyskuje się trzy typy zajętości oktantów i każdemu z nich przyporządkowuje się etykietę [104,110,149,172,174,204], np.:

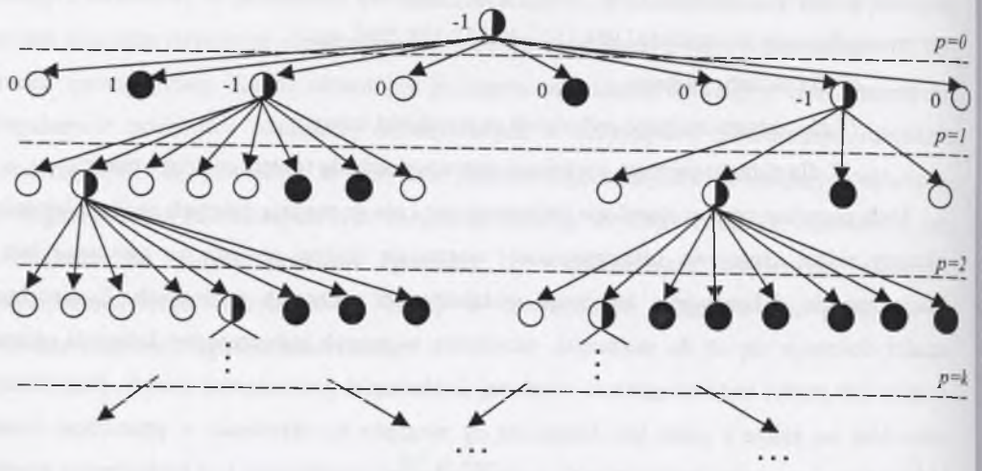
- „*0*" dla oktantu pustego,
- „*1*" dla oktantu zajętego całkowicie przez obiekt sceny,
- „-1" dla oktantu zajętego częściowo przez przeszkodę (oktant nierozpoznany).

Dwa pierwsze typy są określone jednoznacznie i nie wymagają dalszych analiz. Natomiast oktanty zajęte częściowo (nierozpoznane) wymagają dalszej analizy, co związane jest z koniecznością dokonywania kolejnych podziałów na wyższych poziomach. Koniecznych analiz dokonuje się aż do momentu określenia w sposób jednoznaczny kategorii oktantu (zajęty lub pusty) bądź osiągnięcia założonej dokładności prowadzonej analizy. Klasyfikacja oktantów na zajęte i puste jest konieczna ze względu na określenie w przestrzeni wokół robota tzw. przestrzeni bezkolizyjnej, w obrębie której poszukiwana jest bezkolizyjna ścieżka robota. Tworzą ją, co jest oczywiste, wszystkie te oktanty, które posiadają etykietę „*0*". To spośród nich wytypowany powinien zostać poszukiwany zbiór rozwiązań dopuszczalnych, a elementy charakterystyczne oktantów identyfikowane są z poszukiwanymi punktami węzłowymi planowanej bezkolizyjnej ścieżki robota.

Uzyskane w ten sposób dane tworzą graf postaci drzewa (rys. 3.4) [12,69,110,132]. Jego wierzchołkami są środki (lub wierzchołki) analizowanych oktantów, zaś krawędziami odległości pomiędzy nimi (metryka euklidesowa stanowiąca funkcję kosztu heurystycznego). Drzewo to, którego wierzchołki rodzicielskie są oktantami na odpowiednio niższym poziomie podziału (pkt 3.2.2) jest drzewem decyzyjnym metody oktalnej [110], definiującym otrzymaną w wyniku analizy oktalnej przestrzeń poszukiwań (rys. 3.4). Drzewo to, lub jego algebraiczna postać, tj. oktalna macierz incydencji [172,173,174,204], jest podstawą poszukiwania zbioru rozwiązań dopuszczalnych [43,49,132], w przestrzeni którego znajduje się poszukiwaną bezkolizyjną ścieżkę robota (najkrótszy cykl, bez powtórzeń i pętli, określony pomiędzy oktantami z etykieta „*0*" - pusty [43,107]).

Metoda oktalna, mimo swojej niewątpliwej prostoty, charakteryzuje się dość istotnymi wadami:

- Po pierwsze, nie proponuje strategii przeszukiwania, czyli nie definiuje wektora poszukiwań [132]. Typowym sposobem działania planera ścieżek robota wykorzystującego technikę planowania oktalnego jest strategia przypadkowego błądzenia [30,43,68,101,132,147], czyli szukania „po omacku". Stąd też kierunek poszukiwania jest przypadkowy, a to nie ułatwia zastosowania jakiegokolwiek kryterium optymalnościowego poszukiwań. Błądzenie przypadkowe, aby stworzyło



Rys. 3.4. Przykład drzewa decyzyjnego metody oktalnej. Kółka puste w środku oznaczają oktanty z etykietą *0*, zaczernione z etykietą *1*, zaś zaczernione w połowie oznaczają oktany z etykietą *–1*; *p* – poziom podziału oktalnego

Fig. 3.4. Example of a decision tree in view of the octal method. Blank circles denote the octants labeled *0*, obscured circles labeled *1*, whereas partly obscured circles – labeled *–1*; *p* – level of octal division

szansę powodzenia, wymusza konieczność całościowej analizy przestrzeni. Niedogodność ta, jak się wydaje kluczowa, odkrywa kolejne, istotne wady metody oktalnej.

- Po drugie, ze względu na konieczne całościowe rozpoznanie przestrzeni, najczęstszą techniką jej analizy jest dokonywanie kompleksowego podziału równomiernie na wszystkich poziomach podziałów dla wszystkich oktantów (generalnie, metoda dopuszcza dokonywanie podziałów lokalnie, tylko w obrębie wybranych oktantów [12,88,132,204,205]), co upraszcza etykietowanie i obliczanie parametrów geometrycznych oktantów, ale znacząco spowalnia cały proces planowania, który w wielu miejscach jest niepotrzebnie zbyt szczegółowy. Technika ta wynika z ujęcia rastrowego otrzymywanej reprezentacji oktalnej przestrzeni, co przy niskich poziomach podziałów daje zbyt małą precyzję odwzorowania wszystkich elementów sceny. Dlatego, w celu zwiększenia dokładności odwzorowania oktalnego konieczna jest bardziej szczegółowa analiza, a ta prowadzi do znaczącego rozrostu przestrzeni przeszukiwań, reprezentowanej przez drzewo oktalne. Działanie takie jest proste i nie utrudnia orientacji w przestrzeni, gdyż - jak wiadomo - metoda oktalna pozwala na jednoznaczną identyfikację wszystkich sześcianów na podstawie znajomości jedynie

wartości n i b^0. Skutkiem takiego postępowania jest rozrost przestrzeni poszukiwań (drzewa decyzyjnego) w postępie:

$$V = \left(2^3\right)^p \tag{3.2}$$

wraz z każdym podziałem. Dlatego, chcąc wyeliminować, bądź ograniczyć zjawisko przypadkowego błądzenia, wymagane jest zastosowanie odpowiedniej strategii poruszania się planera w środowisku reprezentowanym odpowiednim drzewem oktalnym, tak by tworzone drzewo decyzyjne mogło być możliwie najmniejsze.

- I wreszcie po trzecie, istotnym problemem, którego metoda nie precyzuje, jest poziom zagłębienia drzewa decyzyjnego otrzymywanego w metodzie, czyli poziom p konieczny, a zarazem – dla uproszczenia analizy – możliwie najniższy. Jak łatwo się domyślić, głębokość grafu oktalnego jest wypadkową: wymaganej precyzji ruchu robota (kolejne odcinki planowanej ścieżki robota wyznaczane są pomiędzy kolejnymi oktantami) oraz złożoności geometrycznej samej przestrzeni technologicznej (gęstość rozmieszczenia obiektów technologicznych, wielkość przedmiotu manipulowanego przez robot, wymaganej jakości – gładkości – kształtu ścieżki, itd.).

Z tego też powodu istotnym problemem metody jest sposób efektywnego przeszukiwania powstałego drzewa decyzyjnego. Dla zadań NP-trudnych (eksponencjalnych [43,56,85,171]), a takim jest omawiany proces planowania w przestrzeni charakteryzującej się przyrostem opisanym zależnością (3.2), liczba wszystkich możliwych do zrealizowania rozwiązań (zadania o charakterze permutacyjnym) może być określona wg zależności Stirlinga [43,151]:

$$N! \cong \exp\left[N \cdot \ln\left(\frac{N}{e}\right)\right] \cdot \sqrt{2\pi N} \tag{3.3}$$

gdzie: N – liczebność zbioru poszukiwań (liczba wszystkich wierzchołków drzewa decyzyjnego metody oktalnej, określona dla równomiernego podziału wszystkich oktantów, jest równa $N = 2^{3p}$).

Omówienie problemu czasu związanego z analizą przestrzeni i wynikającego ze złożoności obliczeń można znaleźć również w pracach [37,57,127,158], a Mansor i Morris [121] uważają, że jest to najważniejszy problem zadania planowania.

Dla tak dużej liczby istniejących potencjalnie możliwości rozwiązań dopuszczalnych dostatecznie sprawnymi algorytmami poszukiwania rozwiązań optymalnych są wspomniane już algorytmy heurystyczne (pkt 3.2.2).

Ze względu na powtarzający się cykl procesu planowania i konieczność powtarzania procesu rastrowania, ogólnie stosuje się dwie podstawowe strategie działania metody [69,110,173,174]. Jest to strategia oparta na:

- przeprowadzeniu każdorazowo podziałów począwszy od oktantu podstawowego ($p=0$) stosownie do wymaganego dokładnością analizy poziomu podziału ($p=k$); proces podziału można zatrzymać na dowolnym poziomie zaspokajającym potrzeby zadania;
- zrealizowaniu jednorazowego cyklu całościowego podziału analizowanej przestrzeni na odpowiednio małe oktanty, aż do pewnego poziomu $p=k$; stosowana jest rzadko, gdyż trudno jest ustalić uniwersalny poziom k końcowego podziału odpowiadający całości realizowanego zadania.

W przypadku:

- pierwszym każdorazowo dokonywany podział podporządkowany jest potrzebom ruchu robota i warunkom procesu planowania, dlatego uszczegółowienie analizy może się odbywać lokalnie, np. tylko w sąsiedztwie obiektów sceny;
- w drugim zaś uzyskuje się drzewo z gałęziami zawsze o tej samej głębokości dla wszystkich oktantów, niezależnie od rzeczywistej potrzeby, co znacznie rozbudowuje powstającą przestrzeń poszukiwań.

Zawsze jednak, niezależnie od strategii, problemem jest określenie dokładności odwzorowanie oktalnego, czyli wymaganego zadaniem poziomu podziału oktalnego. W tej sytuacji wspomniane algorytmy przeszukiwania heurystycznego mają do spełnienia jeszcze jedno istotne zadanie. Mianowicie, powinny one przeszukiwać drzewo decyzyjne tak, by bez zasadnej potrzeby nie penetrować go zbyt daleko w głąb, niezależnie od poziomu dokonanego podziału. W efekcie oktanty w wielu przypadkach (miejscach przestrzeni) praktycznie nie są wykorzystywane, poza miejscami o dużym zagęszczeniu przestrzennym przeszkód.

3.2.2.1.2. Kulowa aproksymacja przestrzeni

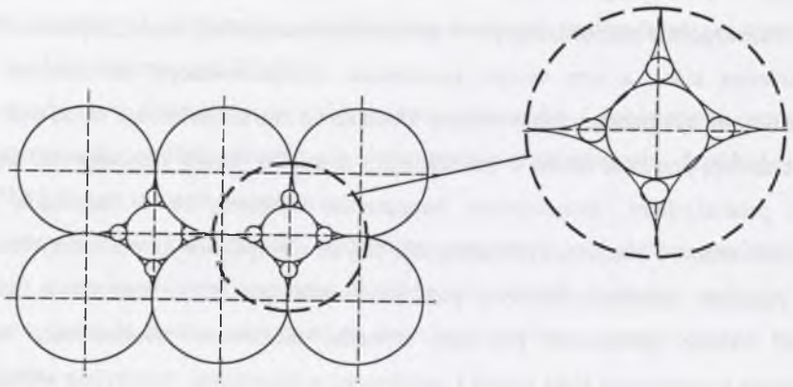
Omówiona metoda drzew oktalnych, jak wskazano, charakteryzuje się rastrowym odwzorowaniem przestrzeni [43,110]. Odwzorowanie to nie spełnia kilku istotnych warunków wymaganych z punktu widzenia funkcjonowania robota. Mianowicie, bardzo ważnym parametrem ruchu robota jest orientacja związana ze zmianą sposobu osiągania

wskazanych przez programistę położeń robota. Zagadnienie to jest tym bardziej ważne, że robot – szczególnie manipulacyjny – w położeniach manipulacji musi „dopasować" orientację przestrzenną kiści, a tym samym przedmiotu manipulowanego, do miejsca, w którym przedmiot ma być pobrany lub odłożony. Dodatkowo zmiana orientacji może być wymuszana koniecznością przejścia robota z przedmiotem pomiędzy blisko znajdującymi się względem siebie przeszkodami. Jednocześnie wspomniano również, że w metodach planowania trajektorii robotów stacjonarnych przyjmuje się, że manipulator robota identyfikowany jest z jego punktem charakterystycznym pozycjonowania (pkt 3.1). Warunki te sprawiają, że podział oktalny przestrzeni powinien stwarzać możliwość zrealizowania wymaganych warunków orientowania kiści robota i przedmiotu w przestrzeni. Intuicyjnie wynika z tego, że najprościej warunek ten mógłby być spełniony, gdyby bok oktantu b^p był nie mniejszy niż średnica d kuli opisanej na przedmiocie manipulowanym, czyli by zachodził warunek:

$$b^p \geq d \tag{3.4}$$

tym samym możliwe byłoby we fragmencie przestrzeni zajętej przez nią dokonanie orientacji we wszystkich trzech osiach określających kąty Eulera [43,69,102,110,132].

Oczywiście, warunek ten nie zawsze może być spełniony, gdyż stosunkowo duże wymiary gabarytowe przedmiotu manipulowanego przekraczają wymaganą dokładność odwzorowania oktalnego, które wymusza dokonywanie bardziej szczegółowych podziałów, choćby tylko w wybranych miejscach. Bajcsy, Mohr i Solina [69,110,132] oraz inni [22,164] proponują rozwiązanie tego zagadnienia przez aproksymowanie analizowanej przestrzeni kulami (sferami) (rys. 3.5). Jednakże, istotnym problemem metody jest określenie długości promieni kul, jakimi należy wypełnić analizowaną przestrzeń, podobnie jak wartości b^p w metodzie oktalnej. Jak łatwo zauważyć, wykorzystanie do tego celu kul o promieniu określonym warunkiem (3.4) wprawdzie umożliwi swobodną orientację robota w przestrzeni (elementem charakterystycznym takich rastrów są środki powstałych kul), jednak skutkiem ubocznym takiego działania będą obszary puste (pomiędzy kulami), które nie są objęte analizą. Dlatego też w metodzie proponuje się wypełnienie tych obszarów kulami o odpowiednio mniejszym promieniu, stosownie do wielkości obszarów pustych (rys. 3.5). Mechanizm ten jedynie zmniejsza wielkość obszarów nie objętych analizą, ale ich całkowicie nie eliminuje.



Rys. 3.5. Przykład upakowania przestrzeni kulami

Fig. 3.5. Example of space filling with spheres

Podobnie jak w przypadku podziałów oktalnych, reprezentacją takiego podziału jest graf (drzewo decyzyjne), w którym środki poszczególnych kul identyfikowane są z jego wierzchołkami, zaś wagi krawędzi określają odległości pomiędzy poszczególnymi kulami.

Omawiana metoda nie spełniła w pełni pokładanych w niej nadziei, gdyż:

- nie umożliwia, w sposób bardziej precyzyjny niż metoda oktalna, pełnego i prostego odwzorowania przestrzeni (technikę tę najłatwiej zastosować do wydzielonej już z przestrzeni roboczej robota przestrzeni bezkolizyjnej, wolnej od przeszkód),
- zawsze pozostawia obszary nieokreślone, znajdujące się pomiędzy poszczególnymi kulami, niezależnie od ich wielkości,
- prowadzi do utworzenia drzewa decyzyjnego o podobnej strukturze jak w metodzie oktalnej i podobnie jak tam konieczne jest posiłkowanie się przy jego analizie technikami przeszukiwania heurystycznego.

3.2.2.2. Metoda probabilistyczna

Występujący w metodach rastrowych efekt przypadkowego błądzenia próbuje się czasami ograniczyć przez zdefiniowanie rozkładu prawdopodobieństwa, rozpiętego na strukturze powstających z każdym podziałem rastrów. Możliwości takie daje tzw. metoda probabilistyczna [43,132]. Wykorzystuje ona efekt regularnej atomizacji przestrzeni na rastry, np. sześciany (metoda oktalna), z których wybiera się w oparciu o utworzony rozkład prawdopodobieństwa te, które dają największe szanse osiągnięcia zaplanowanego celu, wyznaczając tym samym zbiór rozwiązań dopuszczalnych zawierający możliwe do

zrealizowania bezkolizyjne ścieżki robota. Metoda opiera się na budowaniu i analizie grafu składającego się ze zbioru drzew (lasu), definiującego różne, potencjalnie możliwe ścieżki przejścia z położenia początkowego do końcowego. Każdą określa osobny graf. Stąd występujący w metodzie las drzew, który stanowi zbiór uzyskiwanych rozwiązań dopuszczalnych zadania. Metoda rozwiązuje zadanie planowania w dwóch etapach.

W pierwszym etapie analizy poszczególne grafy, definiujące kolejno wytypowane ścieżki, są grafami nieskierowanymi. Na tej podstawie w oparciu o utworzone definicje:

a) funkcji odległości d w przestrzeni konfiguracyjnej (metoda dopuszcza wykorzystanie zarówno konfiguracyjnej wewnętrznej q, jak i zewnętrznej s zadania, w przypadku drugim jest to przestrzeń sceny, określona kartezjańskim układem współrzędnych),

b) spójności grafu V pozwalającej na łączenie wierzchołków grafu w tzw. komponenty,

rozwiązuje się zadanie planowania.

W etapie drugim wierzchołki drzew, identyfikowane z kolejnymi położeniami robota, są przeszukiwane i łączone w komponenty, w celu znalezienia połączenia konfiguracji początkowej robota z zadaną konfiguracją końcową. Etap ten opiera się na następujących działaniach [43]:

1) wybranie przestrzennej konfiguracji bezkolizyjnej s_b wg przyjętej w zadaniu reguły probabilistycznej, określonej pewną dyskretną funkcją gęstości prawdopodobieństwa, opisanej na analizowanej przestrzeni i charakteryzującej określonym się gradientem zmienności,
2) określenie dla wybranej konfiguracji bezkolizyjnej s_b zbioru wierzchołków sąsiednich N_s,
3) sprawdzenie, czy dla wybranej s i dla każdego wierzchołka $w_i \in N_s$ istnieje bezkolizyjna ścieżka t_i łącząca s i w_i.

Wykorzystując funkcję odległości, otrzymane w kroku 3 ścieżki bezkolizyjne kwalifikowane są jako etapy (kolejne kroki) poszukiwanej ścieżki przejścia do wyznaczonego położenia końcowego.

Odpowiednia kwalifikacja znalezionych ścieżek t_i pomiędzy kolejnymi wierzchołkami przeszukiwanego grafu (etap drugi, krok 3) może stanowić istotny mechanizm zmierzający do ograniczenia przestrzeni poszukiwań (bardzo często heurystycznych), jak również określać kryterium optymalizacyjne, np. proste kryterium najkrótszej ścieżki.

Zastosowanie strategii wyboru losowego nie rozwiązuje istotnych problemów przeszukiwania przestrzeni rozwiązań dopuszczalnych. W pewnych przypadkach może doprowadzić do skrócenia czasu analizy grafu rozwiązań dopuszczalnych V, ale może również doprowadzić do tzw. błądzenia przypadkowego [30,101,147]. Jednym z prostych

kryteriów ograniczenia tego zjawiska może być kryterium podwójnej odległości, tzn. przy definiowaniu funkcji odległości pomiędzy wytypowanym wierzchołkiem w_i a wierzchołkiem docelowym (w warunkach rozwiązania zadania w zewnętrznej przestrzeni stanów praktycznie jest posłużyć się normą odległości euklidesowej [18,34]) wygodnie jest określić odległość do kolejnego, wybranego wierzchołka w_{i+1} (z uwzględnieniem warunku spójności)

3.2.2.3. Metoda sztucznego potencjału

Jest to przykład metody geometryczno-analitycznej. Metoda pól potencjałowych zwana jest również metodą sztucznego potencjału [38,43,110,132,220]. Należy ona do grupy metod heurystycznych, wykorzystujących do przeszukiwania otrzymanego grafu rozwiązań dopuszczalnych planer losowy, w którym heurystyczna funkcja przeszukiwania określona jest zmienną losową. Metoda opiera się na założeniu, że każdy z obiektów statycznej przestrzeni roboczej robota posiada sztuczne pole potencjału, które rozpościera się wokół każdego z nich. Pola te są względem siebie niezależne, więc każde charakteryzuje indywidualnie każdy obiekt sceny (przeszkodę robota) niezależnie. Działanie pola potencjału obiektów sceny polega na tym, że każdy obiekt, który znajdzie się w tym polu, podlega „odpychaniu". Wartość potencjału, jakim obiekt oddziałuje na robota, a co za tym idzie - siły odpychającej „generowanej" przez każdą z potencjalnych przeszkód znajdujących się w przestrzeni roboczej robota (obiektów sceny robota), określana jest zależnością [132]:

$$V_{pj} = \begin{cases} C_{pj}\left(\dfrac{1}{L_j} - \dfrac{1}{L_{zj}}\right)^2 & dla \;\; L_j \le L_{zj} \\ 0 & dla \;\; L_j > L_{zj} \end{cases} \qquad (3.5)$$

gdzie: V_{pj} – wartość potencjału określona dla przeszkody j,

L_j – odległość od przeszkody j,

L_{zj} – ustalona (przyjęta) odległość od przeszkody j, na którą oddziałuje potencjał,

C_{pj} – tzw. stopień wzmocnienia potencjału – dodatni współczynnik ustalony indywidualnie dla każdej przeszkody j (każdego z obiektów) znajdującej się w przestrzeni roboczej.

Podobne rozważania można znaleźć w pracach [38,43,110].

Tak ustalona wartość potencjału oznacza, że potencjał zaczyna oddziaływać na robota dopiero wtedy, gdy zbliży się on do przeszkody na odległość mniejszą niż zadana szerokość

L_{zj} jego pola potencjalnego i wraz ze zbliżaniem się do niej wartość V_{pj} znacząco rośnie. Podobnie wyznacza się pole potencjału V_M, jakim oddziałuje m ogniw manipulatora na otoczenie.

Dodatkowo metoda zakłada, że manipulator robota przemysłowego, wykonując ruch w przestrzeni, jest poddany siłom przyciągania, których wartość jest równa [38,132]:

$$V_g(q) = C^g(P_k - q)^2 \qquad (3.6)$$

gdzie: $V_g(q)$ – wartość siły przyciągania wywieranej na ramię robota w położeniu q,

C^g – współczynnik siły przyciągania – dodatnia wartość stała,

P_k – położenie docelowe ścieżki robota,

q – położenia zajmowane w danym momencie ruchu.

Zatem na manipulator robota przemieszczający się z położenia bieżącego i do końcowego k zadanej ścieżki oddziałują wszystkie siły, których wartość wynosi [132]:

$$\sum_{i=1}^{k} V(q_i) = \sum_{i=1}^{k} V_g(q_i) - V_M - V_p = \sum_{i=1}^{k} C''(P_k - q) - \left\{\sum_{j=1}^{m} V_j(q) + \sum_{i=1}^{p} V_k(q)\right\} \qquad (3.7)$$

Aby zatem manipulator zrealizował swoje przejście do celu w możliwie najlepszy sposób, musi zostać spełniony dodatkowy warunek, by:

$$\sum_{i=1}^{k} V(q_i) \to \max \qquad (3.8)$$

Równanie (3.7) jest równaniem ścieżki, jaką musi zrealizować robot, by osiągnąć zadane położenie z jednoczesnym spełnieniem warunku unikania kolizji.

Metoda potencjału wymaga zdefiniowania funkcji aproksymującej ramiona manipulatora bryłami prostymi (np. prostopadłościany), by określić wartość potencjału, jaki „niesie" ze sobą poruszający się w przestrzeni manipulator. Metoda nie pozwala na definiowanie trajektorii optymalnych, gdyż otrzymane równanie bezkolizyjnej ścieżki robota nie jest sparametryzowane czasem.

Interpretacja metody za pomocą dyskretnej funkcji potencjałowej (poprzez określenie gradientu jej zmiany wyznaczającego kolejne fazy działania planera) prowadzi do otrzymania

grafu. Metoda generuje jego wierzchołki począwszy od konfiguracji początkowej, aż do końcowej. Dogodną metodą przeszukiwania tego grafu jest planowanie losowe [43] polegające na losowym wyborze kolejnego miejsca prowadzącego do celu oraz sprawdzenie dla niego i dla drogi prowadzącej do niego z położenia bieżącego warunków kolizyjności.

3.2.3. Inne metody planowania trajektorii w zewnętrznej przestrzeni zadania

Oprócz omówionych metod, na zasadzie opartej na rastrowej analizie przestrzeni 3D (metody grafowe) działa metoda wykorzystująca wielościany wypukłe [29,110,129,132,163, 218]. Zasada jej działania jest taka sama jak innych metod grafowych. Wykorzystuje technikę budowania wielościanów wypukłych za pomocą płaszczyzn. W wyniku takiego działania otrzymuje się bryły lub ich zespoły (kompleksy), pozwalające na wytypowanie obszarów dostępnych (bezkolizyjnych) dla robota. To zaś pozwala na ocenę stopnia ich zajętości, a tym samym określenia możliwości penetrowania przestrzeni przez robota.

Inną wersją technik rastrowych jest metoda modelowania przestrzeni oparta na zbiorze typowych, uznanych za podstawowe, wielościanów, zwanych czasami prymitywami geometrycznymi [43,91,110,132,203]. Technika ta przeznaczona jest głównie do modelowania przeszkód robota poprzez budowanie złożonych brył, w oparciu o przygotowany zbiór brył podstawowych, będących modelami obiektów technologicznych sceny robota. Dzięki mechanizmom wiązania brył podstawowych w kompleksy uzyskuje się możliwość modelowania obiektów technologicznych i definiowania ich położeń. Tym samym określa się obszary przestrzeni zajęte przez przeszkody (przestrzeń kolizyjności robota) i obszary nie zajęte przez nie (przestrzeń bezkolizyjna dostępna dla robota). Dodatkowo metoda umożliwia, na podobnej zasadzie, budowanie oprzyrządowania technologicznego robota (chwytaków, głowic technologicznych) oraz przedmiotów manipulowanych. Tym sposobem, po wydzieleniu przestrzeni bezkolizyjnej możliwe staje się definiowanie ścieżki robota poprzez penetrowanie przestrzeni bezkolizyjnej i budowanie grafu decyzyjnego, podobnie jak miało to miejsce w metodzie drzew oktalnych. Technika ta jest stosowana we wszystkich systemach graficznych programowania robotów typu CAD (COSIMIR VR, IGRIP, eMPowers), wykorzystujących opracowane bazy danych typowych obiektów geometrycznych pozwalających na modelowanie wirtualnego świata robota i wykorzystanie przy programowaniu oceny wizualnej podejmowanych działań.

Oprócz omówionych metod, w literaturze [3,16,17,19,21,43,50,62,77,105,110,122,125, 144,150,168,178,214,219] znaleźć można jeszcze takie metody, jak:

- metoda losowa, podobna do metody probabilistycznej, lecz nie wymagająca zastosowania funkcji gęstości prawdopodobieństwa,
- metody stymulowane biologicznie (sieci neuronowe, algorytmy genetyczne) i logiką rozmytą. Metody te są bliższe rozwiązywaniu zadania planowania z punktu widzenia sterowania robotem.

3.3. WNIOSKI Z LITERATURY

Wśród metod planowania trajektorii (ścieżek) robotów manipulacyjnych w zewnętrznej przestrzeni stanów dominują metody grafowe i heurystyczne. Nie są one wyposażone w żadne mechanizmy (strategie działania), które pozwalałyby na szybkie rozpoznanie otoczenia robota i typowanie zbioru rozwiązań dopuszczalnych. Analiza przestrzeni i ocena proponowanych rozwiązań dokonywane są każdorazowo indywidualnie, dla każdego wyznaczonego punktu końcowego poszukiwanej ścieżki.

Istotne dla poszukiwań optymalnej trajektorii jest rozwiązanie zadania planowania w zewnętrznej przestrzeni zadania robota. Podanie ciągu konfiguracji zewnętrznych robota (zdefiniowanie ścieżki bezkolizyjnej), odpowiadających jego położeniom pośrednim pomiędzy położeniem początkowym a końcowym przy spełnieniu warunku braku kolizji, jest kluczowe dla rozwiązania zadania planowania trajektorii i sterowania robotem.

Wszystkie omawiane metody rozwiązują zadania planowania w zewnętrznej przestrzeni zadania w kilku fazach [69,110]:

- atomizacja (podział) przestrzeni sceny,
- kwalifikacja poszczególnych rastrów w kontekście ich zajętości przez istniejące w przestrzeni obiekty technologiczne – przeszkody robota na: zajęte (niedostępne) – puste (dostępne) – zajęte częściowo (dostępne warunkowo),
- budowa grafu (drzewa decyzyjnego) reprezentującego podzieloną przestrzeń sceny robota,
- zdefiniowanie odpowiedniej dla zadania funkcji heurystycznej, uwzględniającej najczęściej kryterium minimalnej odległości (najkrótszej drogi) do celu i znalezienie zbioru rozwiązań dopuszczalnych zadania,
- wybór ścieżki najlepiej spełniającej postawione kryterium optymalnościowe w zbiorze rozwiązań dopuszczalnych.

Tak zdefiniowane i rozwiązane zadanie planowania nie gwarantuje znalezienia rozwiązania optymalnego w sensie globalnym. Stosowane najczęściej jako kryterium optymalności poszukiwanej ścieżki robota, kryterium najkrótszej drogi (rozwiązanie suboptymalne) stanowi kryterium lokalne, czyli rozwiązujące zadanie optymalizacji ze względu na jeden parametr. Niemniej, sama technika budowania drzewa decyzyjnego, jak i trudności w znalezieniu odpowiednio sprawnej funkcji heurystycznej raczej nie pozwalają na stosowanie innych kryteriów optymalnościowych, takich jak np. czas, a w konsekwencji uwzględniania zjawisk dynamicznych przy definiowaniu funkcji przejścia [177,184,185,186,187]. Dlatego też metody te pozostają przy rozwiązaniu wyłącznie problemu ustalenia ścieżki robota w przestrzeni (łamanej), łączącej wytypowane wierzchołki ustalonego grafu (drzewa decyzyjnego).

Z problemem planowania ścieżki robota w takim ujęciu wiąże się również zjawisko minimów lokalnych [43]. Spowodowane jest ono doprowadzeniem procesu planowania do tzw. ślepego zaułka, skąd wyjście jest bardzo utrudnione, a czasami wręcz niemożliwe [43]. Mechanizm ten spowodowany jest procesem błądzenia bez uwzględnienia globalnych kryteriów wyboru. W efekcie typowane są miejsca, z których nie ma wyjścia w kierunku wyznaczonego miejsca końca planowanej ścieżki, zaś wycofanie się często wymaga określenia dodatkowych kryteriów planowania [43]. Eliminując na etapie przeszukiwania heurystycznego możliwość powtórnego wyboru rastra, uniemożliwia się wyjście ze ślepego zaułka, w którym znalazł się proces planowania. Powoduje to przerwanie procesu planowania bez powodzenia. Innym problemem związanym z przypadkowym błądzeniem są pętle, czyli przypadki, w których raster wejściowy do obszaru przestrzeni objętego ślepym zaułkiem leży bezpośrednio obok rastra wyjściowego (lub jest z nim tożsamy). Stąd wymóg, by poszukiwana w drzewie decyzyjnym ścieżka spełniała warunek cyklu najkrótszego, bez powtórzeń i pętli w sensie teorii grafów [43], pkt 3.2.2.1.1. Fragment wytypowanej ścieżki objętej pętlą nie wprowadza jakościowej zmiany w proces planowania, lecz powoduje jedynie jego wydłużenie i konieczność wyeliminowania pętli. Wszystkie te przypadki planowania wymagają odpowiednich procedur planowania, czyli strategii. Minima lokalne i pętle mogą więc być rozwiązywane na drodze odpowiednio sprecyzowanej i dobranej:

- techniki planowania (minima lokalne), np. drogą dodatkowych kryteriów wyboru kolejnych rastrów (np. wspomniane już kryterium podwójnej odległości, pkt 3.2.2.2),
- strategii przeszukiwania heurystycznego drzewa decyzyjnego (pętle) [16].

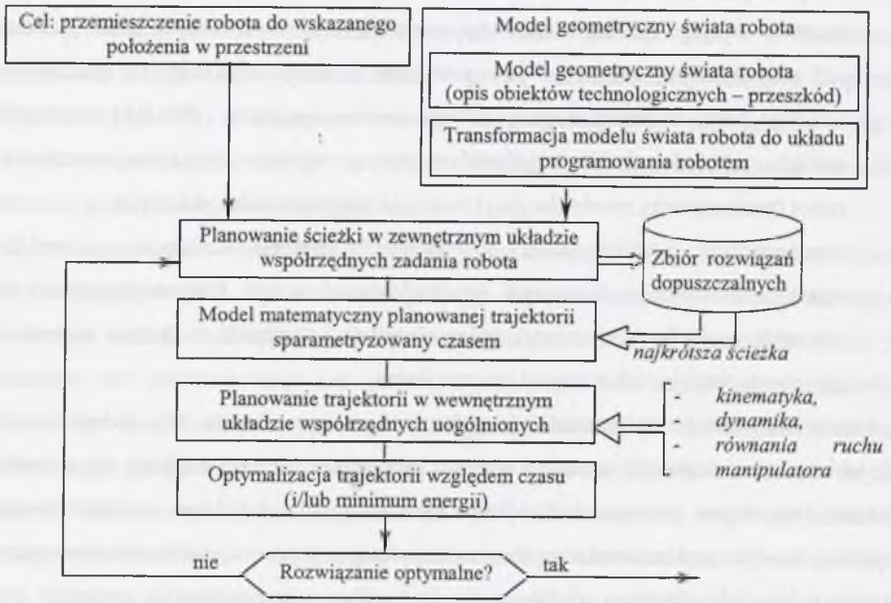
Wszystkie omawiane metody charakteryzują się dodatkowo:

- mechanizmem błądzenia, który czasami próbuje się ograniczyć dodatkowymi warunkami, takimi jak wagi [43,110], czy odpowiednio rozpisaną funkcją probabilistyczną, przez co wzmacnia się skłonności metody do rozwiązań w pożądanym kierunku;
- brakiem pamięci działania, co oznacza brak zdolności metody do przewidywania skutków swojego działania, np. drogą porównywania postawionych zadań z zadaniami już rozwiązanymi (m.in. to jest powodem dokonywania ciągłych podziałów dla każdego nowo wyznaczonego położenia końcowego ruchu robota), by korzystać ze swoich doświadczeń. Niedogodność ta jest szczególnie kłopotliwa, zważywszy że robot (przynajmniej manipulacyjny) realizuje swoje zadania cyklicznie;
- koniecznością wykorzystania dodatkowych strategii działania, pozwalających ograniczyć wielkość przestrzeni przeszukiwania, a tym samym usprawnić proces przeszukiwania heurystycznego, który eliminuje te techniki z obszaru zastosowań w planerach pracujących w czasie rzeczywistym.

Omówione właściwości przedstawionych metod planowania wskazują, że brak jest wśród nich takich, które można by uznać za w pełni sprawne w sensie pożądanej dużej szybkości działania. Dodatkowo, planowanie trajektorii robota w ujęciu globalnym jest możliwe jedynie w rozbiciu na obszar planowania w przestrzeni zadania i w przestrzeni stanów wewnętrznych [43,69,110,132,135]. Efektem tej dwoistości (z każdym z tych obszarów związane są inne metody planowania) zadania planowania jest to, że przyjmowane w nich kryteria optymalizacji poszukiwanej trajektorii są rozłączne, a uzyskiwana ścieżka robota w przestrzeni zadania jest jedynie suboptymalną (w kategoriach długości) - nietrudno bowiem wyobrazić sobie, że ścieżka najkrótsza wcale nie musi być zrealizowana w najkrótszym czasie. Całościowe (globalne z punktu widzenia procesu planowania) rozwiązanie zadania optymalizacji musiałoby być dokonane bowiem w układzie sprzężenia zwrotnego, w którym dla wybranej suboptymalnej ścieżki funkcja przejścia pozwoliłaby na ustalenie optymalnych parametrów kinematycznych i dynamicznych planowanego ruchu robota i tym samym doprowadziłaby do wygenerowania optymalnych parametrów sterowania robotem [43,56,102,177,184,185,186,187]. Dopiero takie całościowe rozwiązanie procesu planowania może zaowocować wyborem trajektorii optymalnej w sensie globalnym [144]. Schemat ideowy takiego procesu pokazano na rysunku 3.6.

Jak wskazano, punktem wyjścia do zadania planowania w przestrzeni stanów zewnętrznych robota (przestrzeń zadaniowa) jest, z jednej strony, wskazane przez programistę

położenie docelowe robota, z drugiej zaś - geometryczny model świata robota. Łatwo zauważyć, że wszystkie omówione metody rozwiązują zadanie planowania, przy założeniu że geometryczny model sceny, a tym samym przestrzeni kolizyjnej i bezkolizyjnej robota jest dany. W istocie żadna z metod nie zajmuje się tym zagadnieniem, z wyjątkiem systemów programowania robotów klasy CAD. Wynika to z dwóch powodów:



Rys. 3.6. Przebieg procedury iteracyjnej optymalizacji planowanej trajektorii robota

Fig. 3.6. Procedure of iterative optimization of robot planned trajectory

- modelowanie otoczenia robota jest zagadnieniem geometrycznie złożonym samo w sobie i wymaga:
  - odpowiednio opracowanych technik definiowania robota i obiektów sceny oraz budowania związków pomiędzy nimi,
  - odpowiednich baz danych i programowych technik obróbki oraz analizy tych danych;
- jest ono metodologicznie niespójne z zadaniem planowania trajektorii; wymaga bowiem innych metod i narzędzi programistycznych i informatycznych.

O ile, w odniesieniu do robotów stacjonarnych, sama technika atomizacji przestrzeni, niezależnie do metody, stosowana jest powszechnie, o tyle główny wysiłek intelektualny w zakresie planowania ścieżek robotów (i to niezależnie czy stacjonarnych i manipulacyjnych, czy mobilnych) koncentruje się na poszukiwaniu odpowiednich strategii działania, pozwalających na znaczne (wymierny w kategoriach szybkości przeprowadzanych analiz) przyspieszenie działania wymaganych procedur przeszukiwania heurystycznego drzew decyzyjnych (grafów), lub ograniczenie wielkości przestrzeni poszukiwań rozwiązań dopuszczalnych. Ten kierunek badań wydaje się w dalszym ciągu aktualny i pierwszoplanowy.

4. Cel, zakres i teza pracy

Jedną z najważniejszych cech metod planowania trajektorii wykorzystywanych w robotyce mobilnej jest ich duża wydajność obliczeniowa. Umożliwia ona rozwiązywanie problemu analizy otoczenia robota mobilnego i generowanie kolejnych fragmentów jego bezkolizyjnej ścieżki technikami on-line [43,60,102,111]. Wynika to z dwóch czynników: płaskości przestrzeni, w której porusza się robot mobilny, oraz odpowiedniej konstrukcji algorytmów planowania. Cech tych nie mają metody analizy przestrzeni, a co za tym idzie - planowania bezkolizyjnych trajektorii manipulacyjnych robotów stacjonarnych. Prostą konsekwencją takiego porównania metod planowania robotyki mobilnej i stacjonarnej jest pytanie: czy jest możliwe wykorzystanie podstawowych walorów metod planowania robotyki mobilnej i zaadaptowanie ich w robotyce stacjonarnej?

Wydaje się, że spośród przedstawionych metod planowania bezkolizyjnych ścieżek robotów stacjonarnych i manipulacyjnych w zewnętrznej przestrzeni kartezjańskiej najlepsza jest metoda oktalna (pkt 3.2.2.1.1) ze względu na swoją przejrzystość, uporządkowanie i proste relacje rekurencyjne, pozwalające na identyfikację każdego powstałego oktantu. Wadą jej jest zaś to, że generalnie wszystkie oktanty są równowartościowe, co skutecznie utrudnia sprawne poruszanie się w rastrowej przestrzeni oktalnej i sprzyja zjawisku błądzenia. Dlatego celowe wydaje się uzupełnienie tej metody o pewną strategię „racjonalnego działania", która pozwoliłaby zintensyfikować powolny proces analizy przestrzeni 3D robota stacjonarnego i manipulacyjnego, przez co można znacząco zwiększyć jej przydatność do praktycznego wykorzystania w procesie programowania off-line robotów przemysłowych. Umiejscowienie omawianego zagadnienia w przestrzeni metod planowania bezkolizyjnych trajektorii robotów, zarówno stacjonarnych, jak i mobilnych, ilustruje tablica 4.1.

Tablica 4.1

Umiejscowienie problematyki badań objętej zakresem pracy

		METODY ANALIZY PRZESTRZENI ROBOTA		
		TOPOLOGICZNE	**GEOMETRYCZNE (GRAFOWE)**	ANALITYCZNO-GEOMETRYCZNE
METODY PLANOWANIA RUCHU ROBOTÓW	**STACJONARNYCH**	[163]	Obszar badań omawiany w pracy Belier i inni, 1991 [12]; Daman i inni, 2002,[35] Dulęba, 2001,[43]; Etzion i inni, 2002,[48]; Jacak, 1991,[69]; **Kost, 2003,[85];** **Kost** i inni, **2003,[88];** **Kost, 1996, [96];** Krishnan i inni, 1996,[104]; Latombe, 1993,[110]; Ngoc-Minh, 2004,[134]; Rafanidis i inni, 2003,[156]; Saramago i inni, 2000,[164]; Skołud B. **Kost G., 1998,[172];** Skołud B. **Kost G., 1993,[173],** Skołud B. **Kost G., 1992,[174];** Ting i inni, 2002,[193] Uptal i inni,1998,[203]; Vörös,2000,[204]; Vörös,2001,[205]	[38,43,110,132,220]
	MOBILNYCH	[15,36,38,61,75,77, 86,111,113,119, 205,135]	[29,43,62,111,132,149]	[43,56,69,111]

Uwzględniając powyższe uwagi, przyjęto następujący cel pracy:

Opracowanie skutecznej metody, pozwalającej na sprawne i szybkie planowanie bezkolizyjnych ścieżek robota przemysłowego poprzez ograniczenie rozmiaru drzewa decyzyjnego, stanowiącego przestrzeń poszukiwania rozwiązań suboptymalnych, przy wykorzystaniu mechanizmu analizy oktalnej, wyposażonego w sprawny system wartościowania i selekcji uzyskiwanych rastrów (oktantów), opartego na stochastycznym procesie decyzyjnym Markowa. Tym samym celem pracy jest stworzenie możliwości obiektywnego, niezwiązanego z intuicyjnym działaniem programisty, wyznaczania bezkolizyjnych dróg robota w przestrzeni sceny, dla systemów programowania off-line robotów przemysłowych.

Zakres pracy obejmuje opracowanie:

1) strategii poruszania się robota w przestrzeni, w oparciu o metodę oktalną i odpowiednio skonstruowany system ocen,

2) algorytmu planowania bezkolizyjnych ścieżek robota w oparciu o metodę oktalną i sterowanie stochastyczne, bazujące na procesie decyzyjnym Markowa

poprzez:

- ustalenie kryteriów określania wielkości oktantu podstawowego w zależności od warunków technologicznych, w jakich pracuje robot (zagęszczenie otoczenia przeszkodami, wielkość przedmiotu manipulowanego),
- odpowiedni system znakowania oktantów, wiążący ze sobą stopień ich zajętości przeszkodami i bliskość ich położenia względem przeszkód,
- zdefiniowanie funkcji oceny typowanych w procesie planowania rozwiązań,
- strategię wyboru najkrótszej ścieżki do położenia docelowego, eliminującą proces błądzenia i pozwalającą tym samym na znaczne zmniejszenie rozmiarów oktalnego drzewa decyzyjnego.

Realizacja postawionego celu i określonego zakresu pracy zmierza do udowodnienia następującej tezy pracy:

W oparciu o odpowiednio skonstruowaną strategię działania i system ocen wykorzystywanych w procesie planowania bezkolizyjnych ścieżek robotów przemysłowych opartym na metodzie oktalnej, możliwe jest znaczące zmniejszenie przestrzeni poszukiwania rozwiązań dopuszczalnych, a tym samym skrócenie czasu pozyskiwania poszukiwanego rozwiązania drogą analiz heurystycznych.

4.1. ZAŁOŻENIA METODY

Chcąc zrealizować przyjęty cel, zakres i tezę pracy, określono podstawowe i ogólne założenia opracowywanej metody:

- zadanie planowania bezkolizyjnych ścieżek robota rozwiązywane będzie dla manipulacyjnych i stacjonarnych robotów przemysłowych, mogących poruszać się na torowiskach identyfikowanych w procesie sterowania robotem z dodatkową osią sterowaną,
- planowanie ścieżki robota będzie się odbywać w przestrzeni zadania robota, zdefiniowanej kartezjańskim układem współrzędnych,

- podstawą opracowanej metody jest wykorzystanie techniki oktalnej aproksymacji przestrzeni objętej sceną robota,
- poruszający się robot identyfikowany jest z punktem charakterystycznym manipulacji (pkt 3.2.2.1.2), związanym z ostatnim elementem ostatniego ogniwa łańcucha kinematycznego manipulatora robota, dla którego rozwiązywany będzie problem kolizyjności,
- w zadaniu pominięty zostanie problem orientacji kiści robota,
- wyznaczoną ścieżkę bezkolizyjną będzie opisywać ciąg współrzędnych kartezjańskich określających jej kolejne punkty węzłowe w przestrzeni wyznaczające poszukiwane rozwiązanie ścieżki bezkolizyjnej do wyznaczonego położenia końcowego,
- poszukiwanie ścieżki bezkolizyjnej będzie oparte na kryterium optymalnościowym, jakim jest kryterium najkrótszej ścieżki łączącej jej położenie początkowe z końcowym.

Przyjmuje się również, że w otoczeniu robota znajdują się obiekty technologiczne i pomocnicze, których wielkość i konfiguracja przestrzenna jest znana, takie jak:

- maszyny technologiczne (np. obrabiarki), które poprzez odpowiednie ruchome kurtyny (drzwi) ograniczają lub udostępniają robotowi ściśle określone fragmenty przestrzeni roboczej,
- wózki (przenośniki) transportujące przedmioty do przestrzeni roboczej robota lub wyprowadzające je stamtąd,
- stanowiska reorientacji i magazyny chwytaków, w których okresowo pojawiają się przedmioty manipulowane (lub chwytaki) i które tym samym również wpływają na wielkość i kształt przestrzeni ruchu robota.

5. Metoda planowania bezkolizyjnych ścieżek dla robotów manipulacyjnych

Z analizy literatury dotyczącej zrobotyzowanych systemów wytwarzania [64,65, 110,111], metod programowania robotów przemysłowych [102,113,115,157,208,209] oraz sposobów rozwiązywania zadań w zrobotyzowanych systemach produkcyjnych przez robota [13,59,113,115,143,144,157,218] wynika, że wszystkie czynności obsługi realizowane przez niego w systemie technologicznym poddawane są ograniczeniom poprzez permanentny i rygorystycznie przestrzegany proces kontroli, koncentrujący się na uporządkowaniu wszystkich czynności. W żadnym zadaniu robota nie ma, z oczywistych powodów, nawet bardzo wąskiego marginesu na indywidualne działanie, wynikające ze zmiany istniejących warunków, jakie mogłyby zajść w jego otoczeniu. Problem sztywnego, czyli niepodatnego na modyfikację, programu robota wymaga jednak spełnienia wielu warunków organizacyjnych, a przy ich niedotrzymaniu często prowadzi do sytuacji kolizyjnej. Sposobem rozwiązania tego problemu jest wyuczenie robota na etapie programowania dokładnie zdefiniowanych interakcji ze środowiskiem. Problem: „jak postąpić dalej?" jest niemożliwy do rozwiązania w przypadku, gdy zaistniała w scenie sytuacja nie mieści się w przewidzianym i ustalonym przez programistę kanonie stanów systemu produkcyjnego i odpowiadających im zachowań robota. W tej sytuacji najczęściej jedyną możliwą i dopuszczalną reakcją robota na wszystkie nieprzewidziane sytuacje jest awaryjne czekanie lub przerwanie programu oraz interwencja operatora, który podejmuje decyzję o działaniu zaradczym. Nawet w systemach robotycznych, określanych jako adaptacyjne [1,3,86,162,192,195], problem przystosowania się robota do nowych warunków organizacyjnych sceny ogranicza się najczęściej do kilku przygotowanych zawczasu sposobów reakcji [86,135], najczęściej typowych, które rozpoznawane są dzięki prostym układom sensorowym rozmieszczonym w otoczeniu robota. Wykryty problem rozwiązywany jest wg prostego i typowego schematu warunkowego [14,86]:

Zasada pracy robota „na zawołanie" (pkt 2.1.1.) powoduje, że robot zmuszony jest przejść do obiektu przywołującego go z różnych miejsc przestrzeni roboczej. Przeoczenie przez programistę systemu (robota) jakichkolwiek, zaistniałych w otoczeniu zmian (np. pozostawionego na stanowisku reorientacji przedmiotu lub przenośnika, może doprowadzić do kolizji. Śledzenie otoczenia prostym, najczęściej binarnym sensorem lub ich zestawem sygnalizującym pojawienie się obiektu w określonym miejscu nie rozwiązuje problemu uniknięcia kolizji bez odpowiedniej, przygotowanej wcześniej ścieżki obejściowej robota. Dodatkowo, istniejące w warunkach przemysłowych silne oddziaływania zakłócające rytmiczność pracy systemu technologicznego (opóźnienia pracy stanowisk produkcyjnych, zakłócenia ruchu robota spowodowane oddziaływaniami dynamicznymi) powodują, że założona w cyklogramie programu robota tolerancja czasowa operacji technologicznych i obsługi musi być właściwie dobrana. W przeciwnym wypadku nawarstwianie się błędów kalkulacji czasowych wszystkich operacji może, przy cykliczności pracy robota, doprowadzić do utraty synchronizacji z otoczeniem, kolizji i zatrzymania całego systemu produkcyjnego. Dlatego jak najbardziej uzasadnione wydaje się stworzenie robotowi takich możliwości, które:

- z jednej strony dałyby mu szansę na elastyczną reakcję w sytuacjach wystąpienia konfliktu w procesie obsługi, przy utracie równowagi czasowej (synchronizacji) pomiędzy otoczeniem a podejmującym działanie robotem,
- z drugiej, pozwoliłyby programiście swobodniej kształtować warunki uporządkowania organizacyjnego systemu,
- z trzeciej zaś, pozwoliłyby wyeliminować element subiektywnej oceny planowanego ruchu robota w przestrzeni przez racjonalizację programowania robota.

Ponieważ robot rozpoczynając obsługę kolejnego stanowiska technologicznego realizuje przejście z jednego miejsca sceny (bieżącego) do wskazanego wezwaniem, to ruch ten musi się odbyć z zachowaniem wszelkich zasad bezpieczeństwa, z których podstawowy jest warunek bezkolizyjności.

5.1. STRATEGIA DZIAŁANIA ROBOTA

Podstawą inteligentnego zachowania się w nieznanym środowisku jest nabyte doświadczenie wynikające z ogólnie przyjętych zasad zachowania, które narzuca standardy

reakcji na występujące w nim sytuacje. Rozumowanie takie, w kontekście omawianych zagadnień, może prowadzić do wniosku, że właściwym podejściem w stosunku do programowanego robota poruszającego się w złożonym geometrycznie otoczeniu byłoby wypracowanie i nauczenie go pewnej ogólnej – uniwersalnej – zasady pozwalającej na kształtowanie jego właściwych (np. bezkolizyjnych) zachowań w tym otoczeniu. Zasada ta, niezależna od realizowanych sekwencji zadań obsługi, powinna pozwolić na racjonalne i bezpieczne (bezkolizyjne) przemieszczanie się robota pomiędzy obiektami sceny do zadanych położeń, niezależnie od jej aktualnej konfiguracji geometrycznej. Jest ona bliska procesowi adaptacji, wynikającemu z potrzeb elastycznego programowania robota i może:

- znacznie ułatwić proces programowania,
- eliminować konieczność wypracowywania przy programowaniu sekwencji „typowych zachowań i reakcji" na zmiany otoczenia w oparciu o możliwe i opracowane scenariusze działania zrobotyzowanego systemu technologicznego,
- pozwolić na znacznie łatwiejsze modyfikowanie zachowania robota, a tym samym większą swobodę programowania, przez typowanie racjonalnie (obiektywnie) uzasadnionych dróg robota w przestrzeni.

Stosunkowo proste podejście do zagadnienia kształtowania reguł racjonalnego działania prezentuje metoda uczenia się ze wzmocnieniem [19,30,105,179,180,211,219]. W założeniach jest ona przeznaczona do zadań, w których wykorzystuje się dynamiczne interakcje podmiotu uczącego się z otoczeniem [1,2,62,130,145,162,169,178,182,192,197, 210,216]. Analizując mechanizm uczenia się ze wzmocnieniem można określić w nim pewne istotne cechy [2,3,11,128,180,182,210].

Po pierwsze: metoda uczy pewnego typu ogólnej reakcji, tzw. strategii, i nie generuje trwałych, określonych procedur postępowania (wyuczonych), z których później uczeń może korzystać, tak jak to ma miejsce np. w metodach uczenia biologicznego (np. sieci neuronowe, algorytmy genetyczne) [19,21,61,125,128,178,182,198,214,219], choć i taki mechanizm reakcji można tu stworzyć w oparciu o rozwiązanie informatyczne zagadnienia (np. odpowiednie bazy oprogramowania). Oznacza to, że typ konkretnej reakcji na zaistniałą sytuację może być każdorazowo inny i zależeć np. od konkretnych czynników, występujących w danym momencie.

Po drugie: metoda wykorzystuje stosunkowo prosty mechanizm korekcji zachowania się ucznia, jakim jest ocena jego postępowania.

Po trzecie: nie wymaga śledzenia otoczenia drogą szybkich interakcji układów sensorowych uczeń-otoczenie, choć dopuszcza taką możliwość. Ważne jest, by istotne ze

względu na zachowanie ucznia zmiany otoczenia mogły być postrzegane, a reakcje układu uczącego się na zachodzące zmiany mogły być efektywne. Zatem środowisko o małej dynamice zmian nie wymaga intensywnego śledzenia, wymaga jedynie, by układ planujący działanie ucznia był poinformowany o jego stanie, przy czym sposób, w jaki się to odbywa i jak często, zależy od specyfiki rozwiązywanego zadania. Śledzenie otoczenia odbywa się zatem drogą okresowych rejestracji jego zmian (po każdym zaplanowanym działaniu ucznia). Informacja o otoczeniu i zachodzące w nim zmiany, będące efektem działań podmiotu procesu uczenia i dostrzeżone przez odpowiedni układ kontroli, są wykorzystywane do planowania kolejnych jego działań. W tym czasie stan otoczenia nie jest analizowany.

Proces prostego uczenia się polega na tym, że uczeń, działając zgodnie z określoną w procesie strategią, otrzymuje za swoje działania ocenę: za reakcję poprawną otrzymuje ustaloną wg odpowiednich kryteriów ocenę dobrą, za złe odpowiednio niższą wg relacji [30,180]:

$$\xi = E\left[\sum_{i=1}^{k} \omega_i \cdot r_i\right] \tag{5.1}$$

gdzie: ω_i – waga uzyskanej nagrody, r_i – nagroda otrzymana za zrealizowanie działania i.

Nagroda ustalana jest każdorazowo dla podjętego przez ucznia działania i (akcji) i na podstawie skutków (zmiany stanu otoczenia), jakie ono wywołało w kontekście postawionego celu. Rolą ucznia jest postępować tak, by realizując postawione przed nim zadanie, osiągnąć jak najlepsze oceny, gdyż jest to równoznaczne z najlepiej wykonanym zadaniem. Efekt nagradzania nosi nazwę wzmocnienia [30,180] i wyznacza tzw. *strategię długoterminowej maksymalizacji nagród* [30,180]. Podjęte działania przebiegają wieloetapowo (iteracyjnie), a maksymalizacja oceny powinna najczęściej być osiągana przy możliwie najmniejszej liczbie iteracji. *Strategią optymalną* tego procesu jest więc ta, która umożliwia osiągnięcie nagrody największej, przy możliwie najmniejszej liczbie podjętych działań [30,180].

5.1.1. Własności metody uczenia się ze wzmocnieniem

Metoda uczenia się ze wzmocnieniem charakteryzuje się kilkoma znaczącymi właściwościami, których atrakcyjność z punktu widzenia potencjalnej metody planowania bezkolizyjnych ścieżek robota może być znacząca. Mianowicie [30,43,85,88,99,100,180]:

- nie wymaga trenowania zachowań ucznia w nowej sytuacji, ani nauczyciela, pozwalającego i ułatwiającego przeprowadzenie uogólnień i wyciągania wniosków z przeprowadzonych przykładów. Oznacza to, że jedynym czynnikiem wartościowania podjętych działań (akcji) ucznia jest aktualna informacja o stanie środowiska, w którym proces uczenia się jest realizowany.
- Nadrzędną strategią procesu uczenia się jest ogólna strategia wartościująca zdobywane oceny, czyli długoterminowa strategia maksymalizacji zdobywanych nagród, różnie interpretowana, w zależności od przyjętych szczegółowych algorytmów, zależnych od specyfiki rozwiązywanego zadania. Wynika z tego, że sposób, w jaki może być ona zrealizowana, nie zależy wprost od *szczegółowej strategii realizacji zadania* (*strategia lokalna*), a sposób, w jaki zostanie to zrealizowane, może być dowolny i wynikać z innych przesłanek, np. ze specyfiki zadania.
- Proces uczenia się następuje drogą kolejno podejmowanych prób i błędów (tzw. błądzenie [9]). Z informacji uzyskanych drogą odpowiednio przeprowadzonej selekcji uzyskuje się potem rozwiązanie optymalne, czyli takie, które maksymalizuje uzyskaną ocenę za osiągnięcie zamierzonego celu działania. Działanie takie jest bliskie procesowi przeszukiwania heurystycznego, np. *BFS* [16,71,72] (pkt 3.2.2). Właściwość ta ogranicza obszar zastosowań techniki uczenia się ze wzmocnieniem tylko do takich przypadków, w których dopuszczalne jest popełnianie błędów.
- Proces realizacji zadania (osiąganie kolejnych stanów zmierzających do osiągnięcia celu) i uczenia się (śledzenia środowiska oraz zdobywania ocen) musi się odbywać naprzemiennie w procesie wzajemnej interakcji, z częstotliwością wymaganą warunkami zadania.
- Metoda uczy pewnego uogólnionego sposobu postępowania (nazywanego ogólną strategią postępowania), jaki należy podejmować w określonych zadaniem warunkach. Nie polega na uczeniu się określonych zachowań przypisanych ściśle zdefiniowanym przypadkom (warunkom zachodzącym w środowisku), co daje znacznie większą swobodę reagowania na istniejące warunki otoczenia.

Proces uczenia się ze wzmocnieniem może być realizowany według różnych algorytmów, przeznaczonych do określania ogólnych strategii postępowania zależnie od warunków środowiska, w jakich przebiega, i czynników charakteryzujących realizowane zadanie. Grupę specjalizowanych *strategii uczenia się ze wzmocnieniem* tworzą algorytmy [21,30,105,130, 180,181,182]:

- *TD* – różnic czasowych,
- *AHC*, oparty na wykorzystaniu agenta pracującego techniką: *aktor-krytyk*,
- *Q-learning*, będący najczęściej stosowanym algorytmem uczenia się ze wzmocnieniem,
- *Sarsa*, blisko spokrewniony z Q-learning i działający wg schematu: *stan-akcja-nagroda-stan-akcja*.

Grupę tę uzupełniają dodatkowe strategie probabilistyczne i licznikowe [30].

Mimo że metoda uczenia się ze wzmocnieniem wykorzystywana jest głównie do planowania zadań w środowiskach postrzeganych jako dynamiczne [1,2,106,130,162,210] (np. roboty mobilne [128,192,216]), to jednak omówione elementy (cechy) mogą być atrakcyjne z punktu widzenia zadania planowania bezkolizyjnych ścieżek robotów manipulacyjnych i stacjonarnych. Za elementy te można uznać:

- prosty i efektywny system oceny planowanych działań i możliwość ich różnicowania za pomocą odpowiednio zdefiniowanych wag ω_i (5.1),
- iteracyjne działanie metody z przewidywalnym w skutkach zakresem planowanych działań (kroków i akcji),
- uczenie ogólnej strategii postępowania, a nie konkretnych reakcji na zachodzące w otoczeniu zmiany.

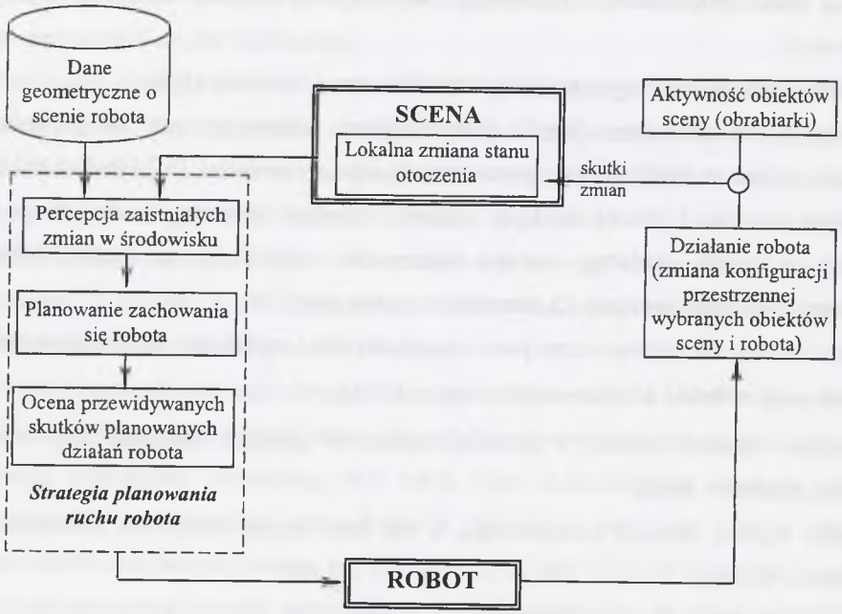
5.1.2. Adaptacja metody uczenia się ze wzmocnieniem do zadania programowania robota

Jak już wspomniano (pkt 3), podstawowym problemem programowania robota jest kreowanie jego ruchu w przestrzeni sceny (zadania). Ze względu na złożoność tego procesu i intuicyjne wartościowanie proponowanych przez programistę rozwiązań, mechanizm uczenia się ze wzmocnieniem mógłby, szczególnie w kontekście metod programowania off-line, umożliwić bardziej racjonalne kształtowanie zachowania się robota w przestrzeni. Szczególnie przydatne mogłoby się okazać wykorzystanie takich elementów metody uczenia się ze wzmocnieniem, jak:

- stosunkowo prosty system nagradzania i definiowania ocen (5.1),
- możliwość budowy ogólnej strategii działania robota w złożonym geometrycznie środowisku technologicznym,

które pozwoliłyby na kształtowanie ogólnego typu zachowania się robota w przestrzeni, szczególnie w kontekście wyznaczanie dróg przejścia do wyznaczonych miejsc w przestrzeni sceny z uwzględnieniem warunków ich kolizyjności. Elementy te mogłyby pozwolić na

wypracowanie ogólnej zasady planowania ruchu robota w złożonej geometrycznie przestrzeni sceny, której nadrzędnym celem byłoby unikanie kolizji. Proces wykorzystujący te dwa elementy mógłby wyglądać tak, jak pokazano na rysunku 5.1.

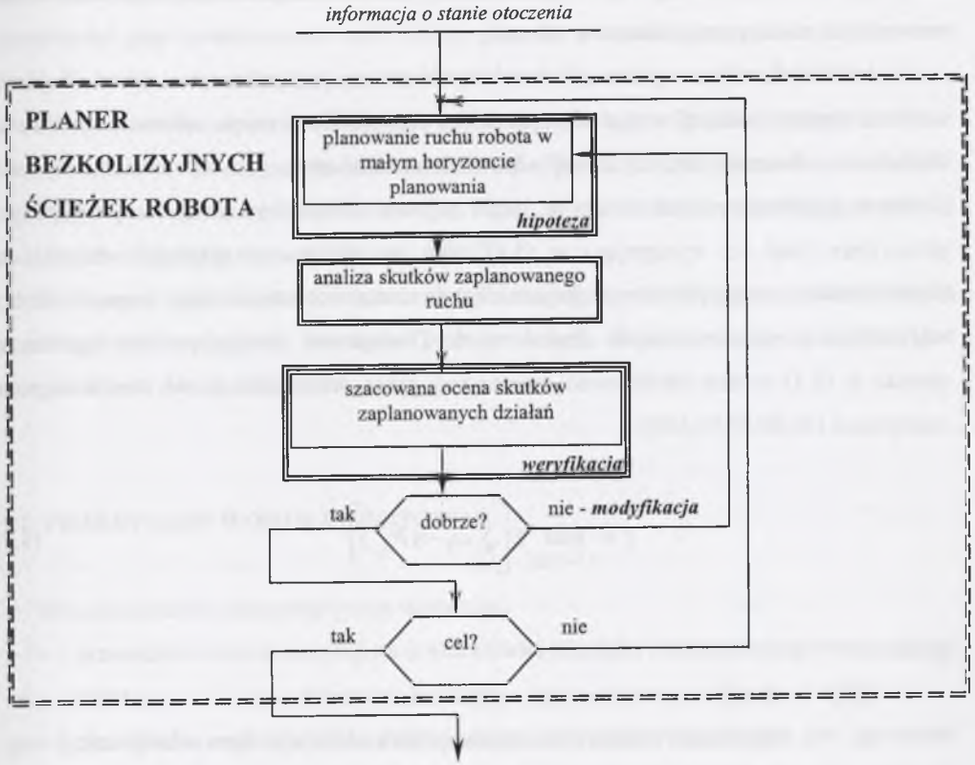


Rys. 5.1. Idea adaptacji mechanizmu uczenia maszynowego dla robota stacjonarnego

Fig. 5.1. Concept of adjusting the machine learning mechanism for a stationary robot

Elementem kreującym proces kształtowania interakcji robota z otoczeniem jest odpowiedni blok (moduł) funkcyjny, zwany *PLANEREM*, z którym robot w procesie planowania ruchu jest ściśle identyfikowany. Zadaniem PLANERA jest więc, bazując na posiadanych informacjach o stanie otoczenia, zaplanować poprawne działanie robota, tak by w oparciu o przyjętą ogólną zasadę planowania ruchu doprowadzić go do wyznaczonego celu. Działanie PLANERA koncentruje się więc na wyznaczeniu takiej ścieżki robota w przestrzeni kartezjańskiej, opisującej przestrzeń wokół robota, by w rzeczywistych warunkach sceny możliwe było jego przejście z położenia początkowego (bieżącego) do końcowego.

Omawiany proces planowania mógłby więc przebiegać wg schematu pokazanego na rys. 5.2. Zatem w oparciu o metodę uczenia się maszynowego ze wzmocnieniem planowanie ruchu bezkolizyjnego robota bazuje na informacji definiującej aktualny stan otoczenia wyrażony zestawem danych geometrycznych, określających układ przestrzenny wszystkich obiektów sceny, co pozwala na wyciąganie odpowiednich wniosków o podjętych działaniach



Rys. 5.2. Schemat działania PLANERA ścieżek robota

Fig. 5.2. Diagram of the operation of robot paths PLANNER

PLANERA, stosownie do założonego celu i przyjętej ogólnej zasady planowania. Ogólna zasada planowania ruchu realizowana przez PLANER koncentruje się na określeniu pewnego typu racjonalnych zachowań robota w środowisku, wiąże ze sobą rodzaj oraz specyfikę działającego w nim robota i jest podporządkowana wypracowaniu planu postępowania robota w swoim otoczeniu, tak by osiągnięcie zaplanowanego położenia końcowego poprzez wszystkie pośrednie było bezpieczne, czyli bezkolizyjne. Dlatego nosi nazwę *ogólnej strategii działania PLANERA* Ψ [40,97,98,106,127].

Działanie PLANERA, ze względu na potencjalne skutki, musi się odbywać na przestrzeni jedynie kilku kroków. Oznacza to, że wypracowana w oparciu o obowiązującą strategię sekwencja ruchów robota powinna swoim zasięgiem obejmować taki ich zakres, by można było przewidywać ich skutki, a tym samym dokonać ich oceny. Planowanie powinno być zatem procesem iteracyjnym (pkt 5.1). Każda z iteracji powinna prowadzić do osiągnięcia

kolejnego stanu pośredniego s_i. Dzięki takiemu działaniu zaplanowana czynność robota prowadzi do zmiany stanu otoczenia robota.

Jednak środowisko, w jakim działa robot, nie zawsze jest jednorodne. Scena obejmuje zarówno obszary puste, tj. wolne od przeszkód, i takie, które są zajęte całkowicie przez nie. Dodatkowo, nieatrakcyjne, czy raczej odpowiednio mniej atrakcyjne są również miejsca o dużym zagęszczeniu przeszkód, tzn. te, gdzie ruch robota musi być realizowany wyjątkowo precyzyjnie. Stąd też występująca w (5.1) waga ω_i powinna uwzględniać wszystkie te czynniki, które ze względu na występujące lokalnie warunki otoczenia mogą w sposób istotny wpływać na ocenę planowanych działań robota. Dlatego też, ujmując problem ogólnie, w oparciu o (5.1) można zdefiniować uzyskiwaną przez poruszającego się robota nagrodę następująco [85,88,93,99,100]:

$$\xi = \max_{i \to \min} E\left[\sum_{i=1}^{k} \omega_i \cdot r(P_{w_i})\right] \tag{5.2}$$

gdzie:

- $E[\bullet]$ - uśredniona (estymowana) wartość zdobywanych ocen,
- ω_i - waga uzyskiwanych ocen określona dla kroku i wg odpowiedniej funkcji wag Ω,
- $r(P_{wi})$ - funkcja nagród uzyskiwanych za zrealizowanie zaplanowanego ruchu do P_{wi},
- i - kolejny zaplanowany krok i.

Składnikami funkcji oceny ξ są zatem funkcja wag, pozwalająca określić dla każdego wytypowanego stanu i przyznanej oceny jej znaczenie, zależnie od lokalnych uwarunkowań miejsca, w którym stan ten został wyznaczony, oraz funkcja określająca wysokość uzyskanej nagrody za osiągnięcie wyznaczonego przez PLANER położenia w kroku i.

Zatem, różnicowanie uzyskiwanych przez PLANER ocen w funkcji warunków środowiska ujmuje określona w zadaniu funkcja wag:

$$\omega_i = \Omega(s_i) \tag{5.3}$$

Tak więc, dysponując określoną funkcją Ω kształtowania wag uzyskiwanych ocen, można tak sterować procesem planowania, by zachowanie robota w środowisku zmieniało się stosownie do warunków, jakie w nim występują.

Jednak znaczącym składnikiem oceny planowanego ruchu robota w otoczeniu są również możliwości jego zrealizowania, czyli $r(P_{wi})$, gdyż nie wszystkie potencjalnie zaplanowane ruchy do kolejno wyznaczonych punktów węzłowych P_{wi}, mogą zostać w jednakowo prosty sposób zrealizowane, a uzyskanie nagrody najlepszej w kolejnym kroku nie zawsze może się okazać możliwe, bowiem, formalnie rzecz biorąc, PLANER może typować kolejne kroki zgodnie, lub niezgodnie z przyjętą strategią Ψ (np. wycofanie się z osiągniętych wcześniej miejsc, lub osiągnięcie stanu osobliwego [43,86]).

By zatem wykorzystać w zadaniu planowania bezkolizyjnych ścieżek robota przedstawiony mechanizm oceniania podejmowanych działań robota, konieczne jest określenie tych dwóch składników oceny ξ.

5.2. PRZESTRZEŃ WOKÓŁROBOTOWA

Przestrzeń robota manipulacyjnego kształtują:

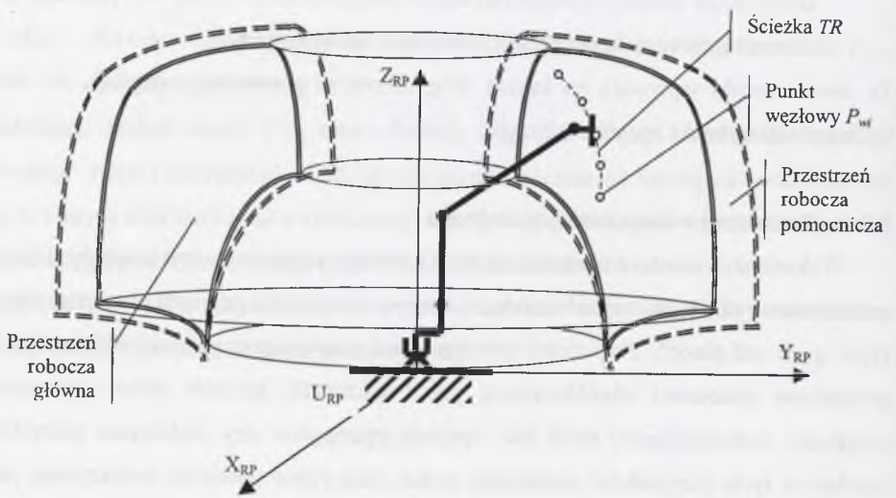
- przestrzeń robocza, wynikająca z możliwości łańcucha kinematycznego manipulatora robota,
- przeszkody, czyli wszystkie obiekty sceny robota, które wypełniając tzw. przestrzeń kolizyjności robota [3,66,85,88,92,93,97,98,129,133,137,164,197] określają wielkość i kształt możliwej do penetracji przestrzeni bezkolizyjnej.

Te dwa czynniki wpływają na kształt dróg robota w przestrzeni, określają ich złożoność (układ przestrzenny) i sposób realizacji.

5.2.1. Przestrzeń robota manipulacyjnego

Wykorzystywane w zrobotyzowanych systemach technologicznych roboty przemysłowe przeznaczone są w głównej mierze do wykonywania szeroko pojętych prac manipulacyjnych typu „pick-and-place". Przy czym przedmiotami manipulacyjnymi są zarówno przedmioty poddawane procesowi obróbkowemu, jak i narzędzia, którymi robot wykonuje swoje czynności technologiczne, takie jak: spajanie elementów czy malowanie natryskowe. W każdym z tych przypadków niezbędną cechą, jaką robot powinien wykazywać, jest duży zakres możliwych do zrealizowania ruchów oraz swoboda ich realizacji, tak z punktu widzenia mechanicznego, jak i programowego. Wynika z tego, że jedną z podstawowych cech, jakie decydują o zakresie aplikacji danego robota, są jego możliwości ruchowe, określone m.in. ruchliwością mechanizmu manipulatora robota. W większości zastosowań

technologicznych możliwości te określone są ruchliwością na poziomie $r_u=5\div6$ stopni swobody, co klasyfikuje roboty posiadające tego typu mechanizm do grupy tzw. robotów uniwersalnych (pkt 2.2). Wyróżniającą cechą takiego mechanizmu jest swobodna penetracja przestrzeni wokół robota i możliwość zrealizowania od 5 do 6 współrzędnych ruchu robota. Oznacza to, że dla typowych przypadków układ robota o takiej liczbie stopni swobody jest robotem nieredundantnym [80,85,144,171], co znacznie ułatwia jego proces sterowania. Konsekwencją takiego stanu jest możliwość realizowania, zależnie od możliwości jego układu sterowania, ruchów interpołowanych: liniowych, kołowych czy interpolacji *TCP* (*Tool-Centered-Point*) [86]. Kolejną cechą takich struktur kinematycznych jest to, że najczęściej są one klasy *5R* lub *6R*, czyli zawierają wyłącznie pary kinematyczne obrotowe (rotacyjne), rzadziej spotyka się struktury z parami translacyjnymi (przesuwnymi), czyli struktury typu np. *5R1T*. Przypadek taki najczęściej ma miejsce wtedy, kiedy robot posadowiony jest na wózku poruszającym się po torowisku i jest to traktowane jako szósta oś sterowana robotem. Konsekwencją tego, że w łańcuchach kinematycznych robotów manipulacyjnych dominują węzły rotacyjne, jest określenie ich struktury mianem struktur sferycznych (czasami określa się je również mianem przegubowych) [65,66] - rys. 5.3. Oznacza to, że typowym ruchem punktu charakterystycznego pozycjonowania robota jest



Rys. 5.3. Układ kinematyczny manipulacyjnego robota przemysłowego

Fig. 5.3. Kinematical system of manipulative industrial robot

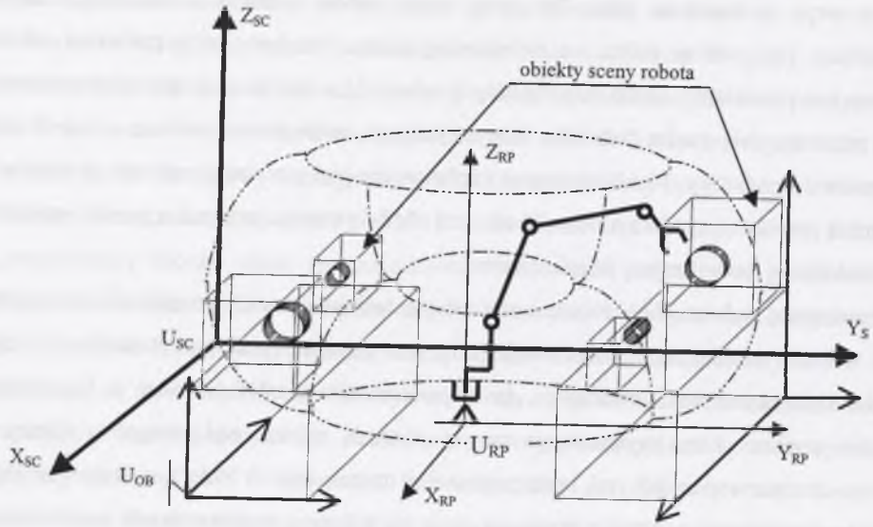
ruch po torze w kształcie łuku. Do opisu ruchu robota zostanie zdefiniowany układ przestrzenny, jaki tworzy robot z przestrzenią sceny. Przyjęto, że z podstawą robota związany jest prostokątny układ współrzędnych robota U_{RP}, taki że jego początek związany jest z pierwszą osią ruchu łańcucha kinematycznego manipulatora robota, a oś Z jest prostopadła do podstawy. W układzie tym zdefiniowana jest przestrzeń robocza, stanowiąca zewnętrzną przestrzeń zadania robota. W nim też zdefiniowane są wszystkie punkty węzłowe P_{wi} poszukiwanej bezkolizyjnej ścieżki robota.

Przestrzenią roboczą Π_{RP} robota nazywać się będzie wycinek przestrzeni otaczającej robota, w której może on realizować swoje ruchy przestrzenne wynikające z możliwości jego łańcucha kinematycznego. Oznacza to, że w przestrzeni tej, zdefiniowanej w U_{RP}, każde położenie punktu charakterystycznego pozycjonowania robota związanego z kiścią, a określone trzema współrzędnymi kartezjańskimi i trzema kątami Eulera opisującymi jego orientację przestrzenną, może być transponowane na wektor q współrzędnych uogólnionych robota na podstawie jego modelu kinematycznego.

Dla robota nieuzbrojonego w narzędzie technologiczne przestrzeń roboczą nazywać się będzie *główną*, a *pomocniczą* - przestrzeń będącą w zasięgu robota uzbrojonego w narzędzie technologiczne (rys. 5.3).

W systemie technologicznym, kiedy robot współpracuje z urządzeniami technologicznymi, programowany ruch robota musi być ruchem bezpiecznym, czyli musi być definiowany tak, by zachodził w przestrzeni nie zajmowanej przez obiekty, które się w niej znajdują (rys. 5.4). W tym celu przestrzeń zajmowaną przez zrobotyzowany układ technologiczny opisuje kartezjański układ sceny robota U_{SC}, w którym znajduje się robot (U_{RP}) i obiekty sceny zdefiniowane w swoich własnych, lokalnych układach współrzędnych U_{OB^j}.

Proces wyznaczenia bezpiecznej ścieżki robota można zidentyfikować jako przejście z jednego stanu s_1, określającego początkową konfigurację przestrzenną manipulatora robota, do stanu s_k związanego z wyznaczonym położeniem końca planowanej ścieżki. Przejście pomiędzy tymi stanami odbywa się poprzez osiągnięcie kolejnych *stanów pośrednich* s_i, identyfikowanych z kolejno wyznaczonymi punktami węzłowymi P_{wi}, określonymi w U_{RP}. Każdy osiągnięty stan s_i robota jest jednoznacznie identyfikowany z punktem węzłowym P_{wi}, dla którego został on osiągnięty $s_i=P_{wi}(x_{wi},y_{wi},z_{wi})$, co można zapisać jako $s_i(P_{wi})$. Ścieżkę TR robota w przestrzeni można więc określić jako:



Rys. 5.4. Układ przestrzenny sceny robota manipulacyjnego

Fig. 5.4. Spatial system of manipulative robot scene

$$TR=(s_1,s_2,s_3,\ldots,s_i,\ldots,s)$$

Zatem, *ścieżką TR* robota w przestrzeni zadania nazywać się będzie krzywą (lub łamaną) określoną w układzie U_{RB}, wzdłuż której przemieszcza się punkt charakterystyczny pozycjonowania robota w trakcie jego ruchu.

W odniesieniu do robota manipulacyjnego, zależność ta jest prawdziwa, jeżeli zachodzi warunek, że:

$$T_Q^{-1}(q_i) = P_{wi} \text{ oraz } T_Q(P_{wi}) = q_i$$

gdzie: T_Q – transformacja prosta kinematyki robota, T_Q^{-1} – transformacja odwrotna modelu kinematycznego manipulatora, q_i – wektor współrzędnych uogólnionych robota określony w *przestrzeni stanów wewnętrznych* (*konfiguracyjnych*) *robota* Q $(\bigcup_i q_i = Q)$.

Osiągnięcie więc jakiegokolwiek położenia robota w zewnętrznej przestrzeni kartezjańskiej (stanu) wiąże się ze zrealizowaniem przez niego odpowiedniego wektora

współrzędnych konfiguracyjnych q_i, któremu odpowiada stosowne przemieszczenie na osiach łańcucha kinematycznego manipulatora.

Zbiór wszystkich stanów s_i zdefiniowanych w zewnętrznej przestrzeni zadania nazywać się będzie *przestrzenią stanów zewnętrznych*: $S = \bigcup_i s_i$.

5.2.2. Ścieżka robota w przestrzeni stanu

Na etapie programowania robota jego ścieżka jest zdefiniowana parametrami geometrycznymi (miejscem docelowym, jakie robot powinien osiągnąć, i sposobem, w jaki to powinien zrealizować) i kinematycznymi (prędkość ruchu). Parametry geometryczne wyznaczające ścieżkę, wg której robot osiąga położenie docelowe, określają również kolejne stany pośrednie s_i, w jakich powinien się znaleźć robot zmierzający do osiągnięcia zadanego stanu końcowego s_k. Konieczność unikania kolizji wymusza zaś realizację zadania przejścia robota z położenia początkowego do końcowego planowanej ścieżki metodą kolejnych *przyrostów* (inkrementów), rozpiętych pomiędzy wyznaczonymi punktami węzłowymi P_{wi}. Przyrosty te nazywać będziemy krokami t_i. Możemy więc ogólnie zapisać, że:

$$s_{i+1}=s_i+t_{i+1}$$

Jak można zauważyć, wielkość tych kroków decyduje o szczegółowości i precyzji uzyskiwanej ścieżki. Stany pośrednie s_i, $i=2,\ldots,k-1$ wyznaczają położenia pośrednie robota leżące pomiędzy początkiem s_1 a wskazanym końcem s_k planowanej ścieżki robota. Oznacza to, że planowana ścieżka robota jest „sumą" następujących po sobie, uporządkowanych liniowo kroków t_i, po zrealizowaniu których robot każdorazowo osiąga kolejny zaplanowany punkt węzłowy P_{wi}, a tym samym kolejny stan s_i. Stany s_i osiągane są drogą realizacji przyjętego sposobu działania δ_i. Liczbę osiąganych stanów pomiędzy stanem początkowym a końcowym: $i=1,2,\ldots,k$ nazywać się będzie horyzontem planowania.

Zatem, zadanie planowania ścieżki robota w przestrzeni stanów zewnętrznych sprowadza się do zagadnienia wyboru odpowiedniego ciągu kroków t_i, w wyniku czego osiągane są kolejne stany pośrednie (elementarne) s_i.

Przyjęto założenie, że zadanie planowania bezkolizyjnych ścieżek robota (zadanie planowania) realizowane jest w środowisku, w którym część obiektów nie zmienia swojego położenia (obrabiarki, magazyny itp.), zaś część zmienia je zgodnie z warunkami procesu technologicznego (np. przedmioty manipulowane, drzwi obrabiarek środki transportowe).

Wprowadza się następujące definicje:

Definicja 5.1.

Krokiem elementarnym t_i ścieżki robota nazywa się dyskretną wartość jego przemieszczenia, zrealizowanego pomiędzy kolejnymi (sąsiednimi) punktami węzłowymi P_{wi} (stanami s_i) wyznaczonymi wektorem przemieszczenia $\vec{d}_{i+1}$, leżącymi w odległości $d_{i+1} = |\vec{d}_{i+1}|$, wyznaczonymi przez PLANER w zewnętrznej przestrzeni robota (przestrzeni zadania) określonej kartezjańskim układem współrzędnych.

Zrealizowanie kroku elementarnego t_i prowadzi do zmiany stanu robota z s_i do s_{i+1}. Kolejne stany s_i, $i=1,2,...,k$ związane są z punktami węzłowymi P_{wi} poszukiwanej ścieżki robota. Krok elementarny t_{i+1} jest zatem związany z drogą d_{i+1} do kolejnego P_{wi+1}, jaką musi zrealizować robot, by osiągnąć kolejny stan s_{i+1}.

Definicja 5.2.

Akcją (przejściem, działaniem) robota a_{i+1} nazywa się jednokierunkowe działanie prowadzące do zmiany konfiguracji przestrzennej robota ze stanu s_i do s_{i+1} na drodze d_{i+1}, w ramach jednego kroku t_{i+1}. Kierunek przejścia określa wektor przejścia $\vec{d}_{i+1}$ oraz droga $d_{i+1} = |\vec{d}_{i+1}|$ Akcja jest zdefiniowana funkcją przejścia δ_{i+1} pomiędzy kolejnymi stanami s_i i s_{i+1}:

$$a_{i+1} : \delta_{i+1} = s_i + t_i$$

Definicja 5.3.

Zadaniem robota nazywa się najkrótszy ciąg kolejnych akcji a_i, tworzących logiczną całość, określony odpowiednimi parametrami początku i końca:

$$\Gamma = \min_i \bigcup_{i=1}^{k-1} (t_{i+1} \cap a_{i+1}) \qquad (5.4)$$

gdzie: $\cap$ - operator koniunkcji.

Zadanie Γ będzie identyfikowane z omawianym zadaniem planowania bezkolizyjnych ścieżek *TR* robota. Jeżeli zadaniem PLANERA (5.4) jest znalezienie bezkolizyjnej ścieżki *TR*

robota w przestrzeni stanów zewnętrznych *S*, to warunek $i \rightarrow min$ można identyfikować z kryterium najkrótszej drogi.

5.2.3. Przeszkody – obiekty sceny

Jednym z głównych problemów każdego zadania planowania ruchu robota w przestrzeni jest określenie sposobu opisu środowiska robota. Potencjalnymi przeszkodami poruszającego się robota w przestrzeni są wszystkie obiekty sceny, z którymi on współpracuje. Chcąc dokonać pełnej analizy tej przestrzeni dla potrzeb rozwiązywanego zadania, konieczna jest znajomość możliwie pełnej informacji dotyczącej tych przeszkód.

Przeszkody robota OB^j można zdefiniować jako bryły utworzone z *i* prostopadłościanów PR^u [29,43,69,92,93,97,110,133]. Każdy z nich jest wyznaczony przez zbiór punktów, określających jego wierzchołki $H(h_g^u)$, gdzie *g* – liczba wierzchołków prostopadłościanu. Te zaś zdefiniowane są zbiorem przecinających się odpowiednich krawędzi $G(g_l^u)$, gdzie *l* – liczba par przecinających się krawędzi.

Krawędzie wycinają z płaszczyzn przechodzących przez nie ściany θ, każdego z prostopadłościanów PR^u. Wszystkie ściany θ, będące czworobokami wypukłymi, tworzą zbiór $T(\theta_e^u)$ (*e* – liczba ścian prostopadłościanu PR^u). Każdy z obiektów jest zdefiniowany w swoim własnym układzie współrzędnych U_{OB^j}, którego położenie jest jednoznacznie określone w układzie sceny U_{SC}. Zatem każdy z obiektów OB^j sceny zdefiniowany jest zależnością:

$$OB^j = U_{OB^j} : (\bigcup_u PR^u : \bigcup_e \theta_e^u,\ e = 1,2,.....,\ u = 1,2,.....)$$

Warunki takiego definiowania obiektów sceny spełniają systemy CAD (modele płaszczyznowe), w szczególności robotowe systemy programowania graficznego off-line (typu CAD: COSIMIR Professional, COSIMIR VR, eMPowers, WorkSpace, IGRIP).

5.2.4. Przestrzeń robocza robota

5.2.4.1. Przestrzeń bezkolizyjna

Przestrzeń, w której możliwe jest zdefiniowanie i zrealizowanie ścieżki robota, musi być obszarem wolnym od obiektów będących dla niego przeszkodami. Zatem, jeżeli przestrzenią

zajmowaną przez zrobotyzowany systemem technologiczny będzie przestrzeń Π_{SC}, a przestrzenią zajmowaną przez przeszkody OB^j robota – przestrzeń Π_{OB}, wtedy [76,97,98, 99,136]:

$$\Pi_w = \Pi_{SC} - \Pi_{OB} \tag{5.5}$$

Przestrzeń Π_{OB} wyznaczają przestrzenie zajmowane przez wszystkie obiekty sceny, więc:

$$\Pi_{OB} = \Pi_{OB^1} + \Pi_{OB^2} + ... + \Pi_{OB^n} = \bigcup_{j=1}^{n} \Pi_{OB^j} \tag{5.6}$$

gdzie: Π_{OB^j} – przestrzenie zajmowane przez poszczególne obiekty sceny, które mogą stanowić przeszkodę w swobodnym ruchu robota, j – liczba obiektów w scenie.

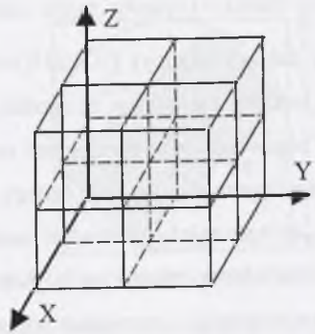
Przestrzeń Π_{OB} będzie nazywana przestrzenią kolizyjności robota, a przestrzeń Π_w przestrzenią bezkolizyjną. Ścieżkę robota, możliwą do zrealizowania w przestrzeni bezkolizyjnej, nazywa się ścieżką bezkolizyjną. Przy uwzględnieniu (5.4) i (5.6) warunek bezkolizyjnej ścieżki robota zachodzi, gdy:

$$\Gamma \cap \Pi_{OB} = \forall_i \left(t_i \cap \delta(s_i, s_{i+1})\right) \cap \forall_j \left(\Pi_{OB^j}\right) = 0 \tag{5.7}$$

co oznacza, że pomiędzy planowaną ścieżką a obiektami sceny nie zachodzi żadna relacja incydencji.

5.2.4.2. Analiza przestrzeni

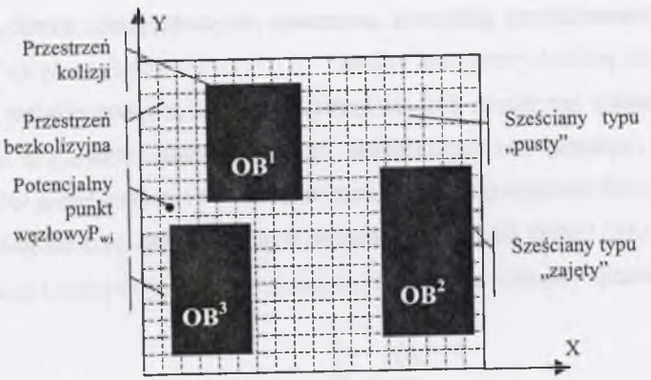
Proponowana metoda planowania zakłada, że analiza przestrzeni robota przeprowadzona jest w oparciu o jej rastrową dyskretyzację na bazie metody oktalnej (pkt 3.2.2.1.1). Zdyskretyzowana scena robota zdefiniowana w swoim własnym układzie U_{SC} i poddana analizie oktalnej na poziomie p charakteryzuje się skończoną liczbą oktantów. Przyjmuje się, że środek geometryczny każdego oktantu identyfikuje go w sposób jednoznaczny i wskazuje punkt węzłowy $P_{wi}(x_i, y_i, z_i)$ planowanej ścieżki robota, a tym samym stan s_i robota po jego osiągnięciu.



Rys. 5.5. Podział przestrzeni roboczej metodą oktalną

Fig. 5.5. Division of operational space by means of the octal method

Dwa oktanty: bieżący i wskazany jako kolejny, określają krok planowanej ścieżki, a odległość ich środków jest równa długości d_i drogi pomiędzy nimi. Tym samym planowany wektor ruchu robota $\vec{d}_{i+1}$, zrealizowany w kroku t_{i+1}, prowadzi do zmiany przestrzennej konfiguracji robota ze stanu s_i na s_{i+1}. Na obszar przykładowej sceny robota nałożono oktalną siatkę sześcianów (rys. 5.5) o boku b^p (3.1), której - dla uproszczenia wywodu - rzut na płaszczyznę X-Y układu współrzędnych U_{SC}, zawierający przykładowe obiekty sceny, pokazano na rys. 5.6. W wyniku dokonywanych podziałów oktalnych jedynie oktanty



Rys. 5.6. Rzut płaski przykładowej przestrzeni sceny podzielonej na sześciany o długości krawędzi b [99]

Fig. 5.6. Plane view of an exemplary scene of the space divided into cubes with edge length b [99]

nierozpoznane wymagają dalszej analizy (oktanty zajęte należące do Π_{OB^j} oraz puste należące do Π_w nie wymagają dalszej analizy) [104,110,149,172,173,174,204]. Problem analizy przestrzeni wokół robota jest tym ważniejszy, że zgodnie z zasadą podziału oktalnego, rozpoczyna się ona od poziomu zerowego, w którym oktant zerowy zawiera całą przestrzeń sceny (pkt 3.2.1.1). Istotnym zatem pytaniem, na jakie należy odpowiedzieć, jest pytanie o poziom podziału oktalnego, do jakiego należy prowadzić analizę oktalną przestrzeni (pkt 3.2.2.1.1), by osiągnąć oczekiwane efekty, jakimi są: z jednej strony, jej uproszczenie i rozległość drzewa decyzyjnego, a z drugiej - uzyskanie odpowiedniej dokładności analizy wyrażonej jednoznacznością kwalifikacji powstałych oktantów w kategorii: zajęty/pusty, z uwagi na to, że wartość t_i decyduje o dokładności uzyskiwanych rozwiązań.

Intuicyjnie wydaje się, że długość krawędzi oktantu b^p powinna być związana zarówno z wielkością przedmiotu manipulowanego przez robot, jak i swobodą jego ruchu, czyli wielkością dostępnej przestrzeni bezkolizyjnej. Spostrzeżenia te potwierdzają dane literaturowe, które zalecają równocześnie, by robot manipulacyjny w zadaniu planowania był identyfikowany ze swoim punktem charakterystycznym pozycjonowania (pkt 3.1) [43,69,110], co zostało zawarte w założeniach prezentowanej metody (pkt 4.1). Wpływają one na tak ważne parametry jakościowe omawianej metody, jak:

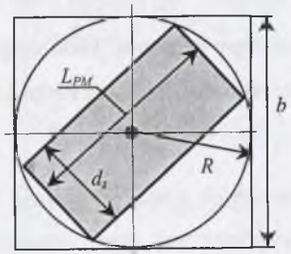
- swoboda orientowania przedmiotu manipulowanego w przestrzeni,
- swoboda poruszania się robota, szybkość ruchu i łatwość omijania przeszkód,
- wielkość przestrzeni poszukiwania rozwiązania (drzewa-grafu) opisującej podzieloną oktalnie przestrzeń roboczą robota.

Uwzględniając wymienione parametry, proponuje się następujący sposób wstępnego szacowania wielkości podziału przestrzeni robota:

a) po pierwsze: należy przyjąć, że długość krawędzi b rastra w siatce oktalnej jest równa średnicy kuli opisanej na przedmiocie manipulowanym (rys. 5.7) [85,88,93]. Przykładowo, jeżeli wielkość przedmiotu manipulowanego o postaci walca określona jest wymiarami L_{PM}xd_s, gdzie L_{PM} – jego długość, a d_s – średnica, to wielkość boku pola elementarnego siatki wynosi:

$$b = \sqrt{L^2 + d_s^2} \tag{5.8}$$

b) po drugie: należy związać konfigurację siatki oktalnej z „zapełnieniem" przestrzeni sceny obiektami, gdyż duża dostępna przestrzeń bezkolizyjna Π_w w stosunku do całej przestrzeni systemu zrobotyzowanego Π_{SC} może doprowadzić do większej swobody



Rys. 5.7. Zasada wyznaczania wielkości pola elementarnego

Fig. 5.7. Principle of designating elementary field size

poruszania się robota i odwrotnie, bardziej zagęszczona przeszkodami przestrzeń Π_{SC} to mniejsza przestrzeń bezkolizyjna i tym samym szansa sukcesu zrealizowania ruchu bezkolizyjnego mniejsza - zatem b powinno być mniejsze. Uwzględniając zależność (5.8), warunek ten można zdefiniować jako:

$$b^s = C_k \cdot b \tag{5.9}$$

gdzie: C_k jest wartością korekcyjną, określającą szanse wystąpienia przestrzeni wolnej od przeszkód, wyrażoną warunkiem:

$$C_k = \frac{\Pi_w}{\Pi_{SC}} = \frac{\Pi_{SC} - \sum_j \Pi_{OB^j}}{\Pi_{SC}} \tag{5.10}$$

Oznacza to, że im mniejszą częścią przestrzeni sceny Π_{SC} jest przestrzeń Π_w, tym bardziej wielkość b^s ulega zmniejszeniu.

Zatem proponuje się, by w pierwszej fazie analizy przestrzeni dokonać jej podziału na sześciany o długości krawędzi równej b^s (5.9). Podział ten nazywany będzie *podziałem startowym*, a sześcian (raster) o boku b^s *sześcianem (rastrem) startowym*. Tak otrzymana długość krawędzi siatki b^s może, zgodnie z metodą oktalną (pkt 3.2.2.1.1), ulegać dalszej zmianie drogą kolejnych podziałów oktalnych, dla których wartość b^p jest teraz równa:

$$b^p = \frac{b^s}{2^p} \tag{5.11}$$

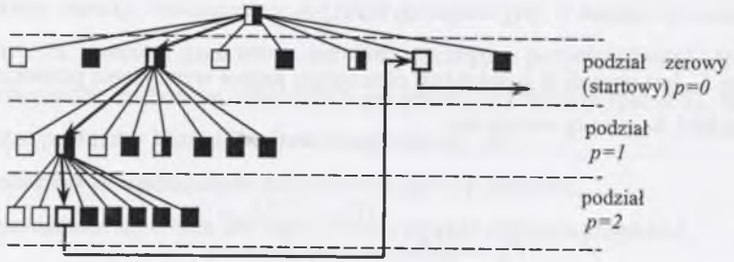
gdzie: p – poziom podziału oktalnego, $p=0,1,2,....$

Wartość b^p będzie nazywana *gradientem planowania*.

Na tej podstawie podział startowy, dla którego $p=0$, można również nazywać *podziałem zerowym*, a raster startowy - *rastrem zerowym*. Oktanty powstałe w wyniku tego podziału p będą nazywane *p oktantami*, a odpowiadające im stany s_i systemu odpowiednio *p stanami* i oznaczane jako s_i^p.

Przy wykorzystaniu właściwości metody oktalnej kolejne podziały dokonywane są wyłącznie dla tych obszarów przestrzeni (tych oktantów), które określone zostały jako nierozpoznane. Spełnienie takiego warunku pozwala znacznie skrócić czas poszukiwania kolejnego rozwiązania dopuszczalnego, gdyż ogranicza dalsze analizy tylko do tych miejsc, które tego bezpośrednio wymagają, a w drzewie decyzyjnym V, reprezentującym analizowaną przestrzeń, rozwijane będą tylko te wierzchołki, które generują p oktanty.

Na tej podstawie z generowanych rastrów można utworzyć graf-drzewo reprezentujące analizowaną przestrzeń, którego postać pokazano na rys. 5.8. Jest to graf skierowany, przy



Rys. 5.8. Przykładowe drzewo V podziału przestrzeni. Linią grubą zaznaczono możliwy przebieg planowanej ścieżki robota

Fig. 5.8. Exemplary tree V of space division. Possible path of robot planned trajectory is marked by the line in bold

czym na tym samym poziomie podziału p odległości pomiędzy poszczególnymi wierzchołkami (identyfikowane ze środkami rastrów) są równe b^p (na poziomie zerowym b^s). Uporządkowanie dokonywanych podziałów pozwala wyznaczyć rekurencyjną zależność, dzięki której możliwe jest ustalenie współrzędnych środka wskazanego oktantu, stosownie do przyjętego planu działania, tzn. [49,73,174]:

$$\begin{cases} x_{ix}^p = \sum_p \left(K^p \cdot b^s + \dfrac{b^s}{2^p} \right) \\ y_{ix}^p = \sum_p \left(K^p \cdot b^s + \dfrac{b^s}{2^p} \right) \\ z_{ix}^p = \sum_p \left(K^p \cdot b^s + \dfrac{b^s}{2^p} \right) \end{cases} \tag{5.12}$$

$$K^p = \begin{cases} 0 & \text{dla oktantów z numerem parzystym na poziomie } p \\ 1 & \text{dla oktantów z numerem nieparzystym na poziomie } p \end{cases}$$

gdzie: b^s, p – jak wyżej, K^p – cecha oktantu wg przyjętego systemu znakowania, ix – numer (indeks) oktantu (pkt. 3.2.2.1.1).

Podobnie można wyznaczyć punkty zaczepienia W (rys. 3.3, pkt 3.2.2.1.1) wszystkich oktantów, jeżeli znany jest tylko poziom podziału p, na którym one powstały, oraz wartość b^s.

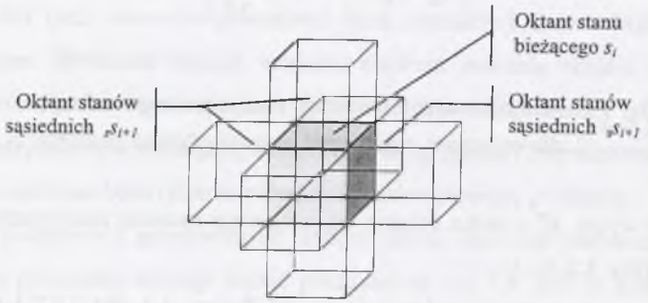
Utworzone i uporządkowane rastry oktalne przestrzeni sceny, zawierające przestrzeń kolizji i bezkolizyjną sceny wraz z robotem, pozostają we wzajemnej, ściśle określonej relacji. Poszczególne rastry, poprzez swoje środki geometryczne, identyfikowane z potencjalnymi punktami węzłowymi P_{wi}, wskazujące potencjalne, możliwe do zrealizowania położenia robota, czyli punkty węzłowe poszukiwanej ścieżki robota, określają kolejne jego stany s_i konfiguracji przestrzennej. Można więc zapisać, że $s_i \in S$. Stosownie do definicji kroku t_i (definicja 5.1, pkt 5.2.1) przejście robota ze stanu $s_i(P_{wi})$ do stanu $s_{i+1}(P_{wi+1})$ może nastąpić wtedy, kiedy stany te są *stanami sąsiednimi* (rys. 5.9) [43,57,110]. Wszystkie stany sąsiednie ${}_s s_{i+1}$ względem stanu bieżącego s_i tworzą *zbiór stanów sąsiednich* [204] ${}_s S = \bigcup_i s_{i+1}$ i ${}_s S \in S$. Liczebność tego zbioru jest równa:

$${}_s N = card({}_s S) = \sum_{p'} \left\{ \left[\left(2^3\right)^{p_{s_i}} - 2 \right] + \left(\frac{2^{3p'}}{2} - 1 \right) \right\} \tag{5.13}$$

gdzie: $p' = p - p_{s_i}$, p_{s_i} - poziom podziału, na którym powstał stan bieżący s_i.

Definicja 5.4.

Stanem sąsiednim ${}_s s_{i+1}$ względem stanu bieżącego s_i, z którego podejmowana jest akcja a_{i+1}, nazywa się taki stan, z którym raster identyfikowany z nim ma bok wspólny (rys. 5.9). Stanowi sąsiedniemu odpowiada stan kolejny s_{i+1}.



Rys. 5.9. Zbiór stanów sąsiednich

Fig. 5.9. Set of contiguity states

Do zbioru stanów sąsiednich, spełniających warunki określone w definicji, należy zatem sześć sąsiednich rastrów (oktantów) o parametrach punktów węzłowych określonych w tablicy 5.1.

Tablica 5.1

Współrzędne punktów węzłowych w zbiorze stanów sąsiednich

	Stany sąsiednie
${}_sS$	$s_{i+1}(x_{i+1}-t_{i+1},y_{i+1},z_{i+1})$
	$s_{i+1}(x_{i+1}+t_{i+1},y_{i+1},z_{i+1})$
	$s_{i+1}(x_{i+1},y_{i+1}+t_{i+1},z_{i+1})$
	$s_{i+1}(x_{i+1},y_{i+1}-t_{i+1},z_{i+1})$
	$s_{i+1}(x_{i+1},y_{i+1},z_{i+1}-t_{i+1})$
	$s_{i+1}(x_{i+1},y_{i+1},z_{i+1}+t_{i+1})$

Dzięki uporządkowaniu stosowanych podziałów, stany s_i są stanami dyskretnymi, opisanymi parametrami geometrycznymi P_{wi}. Dlatego też przestrzeń stanów zadania planowania jest przestrzenią zdyskretyzowaną.

Ponieważ przejście a_{i+1} umożliwia osiągnięcie stanu s_{i+1}, który zawsze jest stanem sąsiednim, dlatego w dalszej części pracy, dla uproszczenia zapisu, przyjmuje się, że ${}_s s_{i+1}=s_{i+1}$.

5.3. PROCESY STEROWANIA STOCHASTYCZNEGO

Jedną z cech metod rastrowych analizy przestrzeni jest zjawisko błądzenia [30,144] (pkt 3.2.2 i 5.1.1), polegające na przypadkowym typowaniu kolejnego stanu ze zbioru stanów sąsiednich. Podjęte działanie PLANERA jest następnie poddawana ocenie w kontekście spełnienia wyznaczonego celu. Jeżeli spełnia oczekiwania, jest akceptowane; jeśli nie - następuje typowanie kolejnego stanu sąsiedniego s_{i+1} zgodne z metodą prób i błędów. Zjawisko to jest podstawą rozrastania się drzewa decyzyjnego V, reprezentującego analizowaną przestrzeń, co w konsekwencji prowadzi do korzystania z algorytmów heurystycznych. Podstawowe zagadnienie, z jakim ma się tutaj do czynienia, jest związane z pytaniem: „którą z możliwych do zrealizowanie akcji a_i powinien podjąć PLANER, aby w jej wyniku osiągnąć stan spełniający najlepiej warunki wyznaczonego celu?" Odpowiedzi na nie mogą udzielić *procesy sterowania stochastycznego* [30,76,108,123,146,147], których jednym z elementów są metody planowania probabilistycznego trajektorii robotów (pkt 3.2.2.2).

Sterowanie stochastyczne jest procesem polegającym na losowym, opisanym pewną funkcją gęstości prawdopodobieństwa rozciągniętą na wszystkie stany analizowanego systemu, sterowaniu obiektem [147]. Funkcja ta ułatwia typowanie kolejnego stanu, stosownie do szans osiągnięcia zaplanowanego celu działania, w oparciu o ustaloną funkcję rozkładu prawdopodobieństwa. Sterowanie stochastyczne ma zatem ograniczyć proces błądzenia [101,147]. Proces ten odpowiada zachowaniu się obiektu sterowanego (w tym przypadku PLANERA identyfikowanego z robotem), który w określonym czasie musi dokonać wyboru kolejnego stanu prowadzącego do wyznaczonego celu. W sytuacji, kiedy PLANER (pracujący wg metody sterowania stochastycznego) ma do dyspozycji pewien zbiór możliwych do zrealizowania stanów (zbiór stanów sąsiednich), musi dokonać wyboru. Zjawisko błądzenia powoduje, że wybór ten jest wyborem przypadkowym ze zbioru stanów sąsiednich, co wynika ze specyfiki konkretnego zadania. Czasami wybór najlepszego stanu może doprowadzić w dłuższym horyzoncie planowania do stanu uniemożliwiającego dalsze działanie PLANERA. Takim przypadkiem jest osiąganie tzw. minimów lokalnych [43]. Niemniej błądzenie, mimo niewątpliwie negatywnego skutku, jakim jest znaczne wydłużenie

całego procesu poszukiwania rozwiązań najlepszych, jest zjawiskiem korzystnym. Funkcja rozkładu prawdopodobieństwa opisująca sposób zachowania się ucznia określa szanse wytypowania najlepszego stanu ze zbioru stanów sąsiednich w perspektywie osiągnięcia wyznaczonego celu. Prawdopodobieństwo to określa zatem szanse zrealizowania postawionego zadania i może być wykorzystane do oceny (parametr $r(P_{wi})$, (5.2)) działania PLANERA typującego kolejny stan s_{i+1}.

W planowaniu bezkolizyjnych trajektorii robotów w procesie sterowania stochastycznego działaniem PLANERA wykorzystuje się *procesy decyzyjne Markowa* [11,15,68,108,116,130, 148,196,211,221]. Stanowią one tym samym model matematyczny całego procesu planowania [30,43,179], a zjawisko błądzenia określane jest tu mianem tzw. błądzenia przypadkowego [68,147].

5.3.1. Procesy decyzyjne Markowa i własność Markowa

Cechą charakterystyczną procesów decyzyjnych Markowa (*PDM*) jest to, że opisują one zjawiska, których stan przyszły zależy wyłącznie od ich stanu bieżącego [68,101,108,147, 148]. Zjawisko to określa się pojęciem „braku pamięci stanów przeszłych" [68,54]. Oznacza to, że informacja o tym, że analizowany układ znajduje się w bieżącym stanie s_1, wystarcza do tego, by określić szanse (prawdopodobieństwo) jego przejścia do kolejnego stanu s_2.

Definicja 5.5.

O ciągu zdarzeń $X(s)_k$ $_0$ (dla uproszczenia zapisywanego jako $X(s)$), który posiada następujące własności [68]:

1) zbudowany jest na skończonym zbiorze dyskretnych stanów S, którego elementy s_i (stany elementarne) są uporządkowane (ponumerowane) w określony sposób, przy czym: $s_i \in S$ oraz $i \in (0,1,...,k)$,
2) na zbiorze S rozpięte jest prawdopodobieństwo wystąpienia poszczególnych stanów s_i o rozkładzie $p(s_i)$,
3) pomiędzy odpowiednimi stanami elementarnymi s_i i s_{i+1} określona jest relacja prawdopodobieństwa przejścia, o wartości $p(s_i,s_{i+1})$,

mówi się, że jest *łańcuchem Markowa* [68,101,108,198], tzn. *wykazuje własność Markowa* [18,68], wtedy i tylko wtedy, gdy zachodzi równość:

$$P(X(s_{k+1}) = s_{k+1} \mid X_k = s_k, X_{k-1} = s_{k-1}, ..., X_0 = s_0) = \\ = P(X(s_{k+1}) = s_{k+1} \mid X_k = s_k) = \\ = p(s_k, s_{k+1}) \qquad (5.14)$$

przy zachowaniu warunku, że $k \geq 0$.

Wartość prawdopodobieństwa $p(s_k,s_{k+1})$ określa prawdopodobieństwo przejścia pomiędzy kolejnymi stanami s_k i s_{k+1} na przestrzeni jednego kroku. Jednakże w łańcuchach Markowa dopuszcza się określenie prawdopodobieństwa zajścia stanu kolejnego na przestrzeni znacznie większej niż jeden krok. Zależność ta określona jest tzw. równaniem na prawdopodobieństwo warunkowe wg Borela [18,34,68]. Dla takiego przypadku zależność taką zapisuje się jako $p(l,s_k,s_{k+l})$, co oznacza prawdopodobieństwo zajścia stanu s_{k+l} pod warunkiem wcześniejszego zajścia stanu s_k, przy czym stan późniejszy osiągnięty został w perspektywie l kroków.

Istotną cechą łańcuchów Markowa jest to, że charakteryzują się one dyskretnością upływającego czasu w trakcie zachodzenia kolejnych zdarzeń. Oznacza to, że każdemu ze stanów elementarnych s_i (będącemu zdarzeniem losowym) przypisany jest pewien określony czas, w którym może on zajść. Prowadzi to do wyróżniania określonych (dyskretnych) jednostek czasu, w których możliwe jest przejście pomiędzy dwoma kolejnymi stanami: s_i i s_{s+1}. Dlatego, z powodu dokładnie określonej zależności czasowej (najczęściej stałej dla wszystkich zdarzeń elementarnych), czas pomijany jest w definicji łańcucha Markowa (5.14) [68,101,108,148], a kolejne zdarzenia mogą być identyfikowane z kolejnymi krokami zachodzącego ciągu zdarzeń. Stąd też tak ważnym elementem definicji łańcucha Markowa jest:

- uporządkowanie wszystkich stanów elementarnych s_i w zbiorze S, z czym wiąże się możliwość ich indeksowania,
- skończoność zbioru S, co oznacza, że $\lim card(S) = N_S$.

Inaczej problem czasu interpretowany jest w przypadku tzw. *procesów Markowa* [18,68,43].

O procesie Markowa $X(t)_{t\geq 0}$ (przez analogię do łańcuchów Markowa oznaczanych jako $X(t)$] mówi się wtedy, kiedy czas zachodzenia kolejnych zdarzeń upływa nie w sposób dyskretny, jak to ma miejsce w łańcuchach Markowa, ale w sposób ciągły. Dodatkowo, czas jest tu podstawowym parametrem następstwa zdarzeń elementarnych. To czas kreuje kolejne zdarzenia, czyli przejście ze stanu s_i do s_{i+1}. Dlatego podana definicja własności Markowa (5.14) dla procesów Markowa wiąże prawdopodobieństwo zajścia kolejnego zdarzenia z czasem, w którym może ono zajść i przybiera następującą postać [18,68]:

$$P(X(t_{c\,k+1}) = s_{k+1} \mid X(t_{c\,k}) = s_k, X(t_{c\,k-1}) = s_{k-1}, \ldots, X(t_{c\,0}) = s_0) = \\ = P(X(s_{k+1}) = s_{k+1} \mid X_k = s_k) = p(t_c, s_k, s_{k+1}) \tag{5.15}$$

gdzie: $X(t_c)$ – ciąg zdarzeń w czasie $t_c \in T$, $t_c \geq 0$; pozostałe oznaczenia i warunki jak dla (5.14).

Konsekwencją uwzględnienia ciągłości czasu w definicji procesu Markowa jest ciągłość w czasie wszystkich jego parametrów. Oznacza to, że zjawiska, które opisuje definiowany dla nich proces Markowa i parametry opisujące jego własności, muszą być interpretowane jako funkcje ciągłe w czasie, a nie dyskretne, jak miało to miejsce dla łańcuchów Markowa. Tym samym, dla procesów Markowa żadna jednostka czasu nie jest wyróżniona w kontekście możliwości zajścia kolejnego zdarzenia. Określone dla łańcucha Markowa prawdopodobieństwa zajścia zdarzenia warunkowego s_i, czy prawdopodobieństwa przejścia z jednego stanu do drugiego, powinny być zdefiniowane jako funkcje ciągłe względem czasu. Dlatego zamiast prawdopodobieństwa wystąpienia stanu s_i mówi się o funkcji prawdopodobieństwa zajścia stanu s_i i wtedy $p(t_c)=p(t_c,s_i)$ [68,147], a zamiast prawdopodobieństwa przejścia występuje funkcja przejść zdefiniowana jako $p(t_c)=p(t_c,s_i,s_{i+1})$ [68,147].

Uwarunkowane ciągłością czasu zachodzenie zdarzeń, a co za tym idzie, konieczność operowania przy ich opisie funkcjami ciągłymi względem czasu[1] powoduje jednak znaczne komplikacje i utrudnienia [68]. Przykładem może być konieczność wykorzystania do definiowania podstawowych parametrów procesu stochastycznego, jakim jest proces Markowa, funkcji prawdopodobieństwa zajścia zdarzenia i funkcji przejść odpowiednich estymatorów funkcji prawdopodobieństw, opartych na całkowych i różniczkowych równaniach stochastycznych [68,147]. Dlatego też Iosifescu [68] zaleca, by dla uproszczenia, tam gdzie jest to możliwe, stosować - bez szkody dla ogólności uzyskiwanych rozwiązań - łańcuchy Markowa, lub ich elementy (wprowadzać element dyskretności opisywanego procesu zdarzeń). Przykładem takiego działania może być dyskretność zbioru stanów elementarnych S, na którym definiowany jest proces Markowa.

Teoria procesów Markowa (łańcuchów również) pozwala na przewidywanie zachowania się analizowanego układu opisanego w czasie. Dzięki prawidłowo zdefiniowanym funkcjom prawdopodobieństw, określonym w definicji procesu w oparciu o ich przebieg w czasie

[1] Należy zwrócić uwagę, że czas jest podstawowym parametrem procesu Markowa [68,147].

wcześniejszym od bieżącego, możliwe jest przewidywanie przebiegu zachodzenia zjawisk, czy zmiany stanów układu, dla czasów następujących po chwili obecnej. Z tego powodu procesy Markowa nazywane są procesami decyzyjnymi Markowa pozwalającymi na przewidywanie i określanie warunków zajścia określonych stanów analizowanego układu w przyszłości [18,68].

Funkcja opisująca przejście (zmianę) układu z jednego stanu do drugiego nazywana jest funkcją przejścia i może być związana z przyjętą przez PLANER ogólną strategią działania Ψ (pkt 5.1.2), lub może wynikać z efektu błądzenia przypadkowego [68,147]. Klasą procesów Markowa wykorzystywanych w procesie sterowania stochastycznego są procesy jednorodne i stacjonarne [68,108,131,146,147,148]. Wprowadza się następujące definicje:

Definicja 5.6.

Proces Markowa nazywa się jednorodnym (skończonym) [68,18], jeżeli spełnia własność Markowa (5.14).

Definicja 5.7.

Jeżeli funkcja $p(s_i)$, określająca prawdopodobieństwo zajścia stanu s_i, nie zależy od czasu, tzn. gdy ma taką własność, że dla każdego odcinka czasu Δt_c, jaki upłynął od chwili t_{ci} ($t_{c\,i+1}=t_{ci}+\Delta t_c$), przejście układu ze stanu s_i do s_{i+1} zachodzi wg tego samego rozkładu prawdopodobieństwa, określonego funkcją $p(s_i)$, to mówi się, że PDM jest procesem stacjonarnym (niezmienniczym) [68,18].

Oznacza to, że znajomość stanu układu w czasie t pozwala wyznaczyć zachowanie się układu w dowolnym przedziale czasowym ($t_c+\Delta t_c$). Wynika z tego, że $p(s_i)$ nie zależy od liczby osiągniętych stanów, lecz wyłącznie od bieżącego i kolejnego stanu układu. Proces Markowa, podobnie jak łańcuch Markowa, również nie wykazuje pamięci stanów przeszłych [54,68].

Określone definicjami 5.5 i 5.6 cechy procesów Markowa mają swój sens jedynie przy spełnieniu warunku, że s_i, jakie może przyjmować analizowany układ, tworzą skończony zbiór stanów $S=S(s_i)$, $i=1,2,\ldots,N$, co jest warunkiem koniecznym skończoności procesów Markowa. Zatem może on zachodzić jedynie w pewnej dokładnie określonej, skończonej przestrzeni probabilistycznej.

Definicja 5.8.

Przestrzenią probabilistyczną nazywa się trójkę (S, F, p) [54,108,147], gdzie:

- S – skończony zbiór stanów systemu ($\bigcup_i s_i \in S$), na elementach którego określone jest prawdopodobieństwo $p(s_i)$ wystąpienia s_i,
- F – wszystkie podzbiory S (np. podzbiory wszystkich m „podstanów" należących do s_i, $\bigcup_m s_i^m = S_i^m \quad i \quad S_i^m = s_i, \quad S_i^m \in S$),
- p – prawdopodobieństwo zajścia stanu s_i, $p=p(s_i)$.

Jak łatwo zauważyć, przestrzeń oktalna (przestrzeń stanów w zadaniu planowania – pkt 5.1.2) może być rozpoznana jako przestrzeń stochastyczna, gdyż:

- skończony zbiór stanów systemu stochastycznego może być zidentyfikowany ze zbiorem stanów systemu planowania,
- podzbiory F zbioru S można zidentyfikować jako podzbiory oktantów powstających lokalnie w wyniku dokonywanych podziałów na poziomie p.

W układzie tym konieczne jest zatem określenie funkcji $p(s_i)$ wyrażającej prawdopodobieństwo zajścia stanu s_i.

W dalszej części pracy, jako proces decyzyjny Markowa będzie się rozumieć wyłącznie procesy jednorodne i stacjonarne.

5.3.1.1. Adaptacja procesu decyzyjnego Markowa do zadania planowania ścieżki robota

Modelem matematycznym procesu sterowania stochastycznego w przestrzeni stanów jest proces decyzyjny Markowa, ergodyczny i stacjonarny. Należy zwrócić uwagę, że taki przebieg procesu sterowania stochastycznego jest stosunkowo bliski procesowi „uczenia się" przez robota zachowania poprawnego w środowisku technologicznym podczas programowania (programowanie samouczące). Programista, próbując znaleźć najlepsze rozwiązanie realizowanego zadania, wybiera losowo takie kolejne konfiguracje przestrzenne robota s_i, które w konsekwencji rokują największe szanse osiągnięcia założonego celu. Jak wiadomo, proces ten przebiega etapowo, a o sukcesie i „nagrodzie", jaką otrzymuje programista za udane zaprogramowanie robota, decydują:

- umiejętność zdefiniowania wszystkich położeń pośrednich (punktów węzłowych) wyznaczających kształt ścieżki,
- łatwość przeprowadzenia robota pomiędzy kolejnymi położeniami P_{wi}.

Wykorzystanym w metodzie mechanizmem uczenia się ze wzmocnieniem jest proces nagradzania planowanych działań robota, realizowanych zgodnie z przyjętą strategią działania PLANERA (5.2). Opierając się na przyjętych w pracy założeniach (pkt 4.1) przyjęto również, że wartość oceny osiąganej w trakcie planowania ruchu robota w przestrzeni sceny zostanie związana z szansą na jego zrealizowanie, wyrażone wartością prawdopodobieństwa, pozwalającego na osiągnięcie wyznaczonego ruchu zgodnie z przyjętą strategią działania. Zrealizowanie szansy zaplanowanego ruchu oznacza:

- wyznaczenie i osiągnięcie kolejnego punktu węzłowego planowanej ścieżki,
- zrealizowanie przemieszczenia z położenia bieżącego do wskazanego, czyli zrealizowanie przejścia pomiędzy stanem bieżącym s_i i wyznaczonym przez PLANER s_{i+1}.

Założenie to powinno umożliwić kształtowanie wymaganej funkcji oceny planowanych działań robota poprzez ujęcie szans zrealizowania zaplanowanego ruchu w postaci funkcji prawdopodobieństwa:

- przejścia pomiędzy sąsiednimi stanami: bieżącym s_i i kolejnym wskazanym w zbiorze stanów sąsiednich ${}_s s_{i+1}$,
- osiągnięcie kolejnego stanu ${}_s s_{i+1}$.

Powinno to umożliwić wygenerowanie przez PLANER, w oparciu o schemat działania określony na rys. 5.2, bezkolizyjnej ścieżki robota, co może znacznie ułatwić proces jego programowania, w szczególności na etapie TPP przez systemy off-line, poprzez wytypowanie bezkolizyjnej ścieżki robota, posiadającej znamiona ścieżki najkrótszej (proste kryterium optymalnościowe) pozbawionej uznaniowego (wg programisty) charakteru.

Dulęba [43] podaje, że problem ciągłości czasu, wymagany w PDM, w robotyce wynika nie z jego upływania w sensie ogólnym, lecz raczej z faktu, że w procedurze sterowania robota „na żądanie" fakt wystąpienia zdarzenia jego przywołania do stanowiska wymagającego obsługi może zajść w dowolnej, precyzyjnie trudnej do określenia chwili. Tym samym czas w analizowanym zadaniu nie musi spełniać roli argumentu w funkcji opisującej planowaną, bezkolizyjną ścieżkę robota, gdyż zgodnie z założeniami rolę tę spełniają parametry geometryczne poszukiwanej ścieżki robota. Fakt wystąpienia jakiegoś zdarzenia sygnalizowany jest tylko modyfikacją danych, opisujących aktualną konfigurację geometryczną analizowanej sceny. Pozwoli to wykorzystać zalecenie Iosifescu [68], by tam, gdzie jest to możliwe stosować w rozwiązywaniu postawionego problemu elementy metodyki łańcuchów Markowa (pkt 5.3.1). Założenia takie pozwolą znacznie uprościć konieczne analizy. Przyjmując dodatkowy warunek, że PDM jest stacjonarny i jednorodny można

wyeliminować czas z funkcji prawdopodobieństw warunkowych PDM. Konsekwencjami tych założeń są:

- zbiór S określający przestrzeń stanu jest skończony, niezależnie od dokonanego podziału,
- każdy element $s_i \in S$ jest dyskretny, i na kolejnych podziałach tworzy również dyskretnie zdefiniowane p stany,
- warunkiem wystarczającym, by funkcje prawdopodobieństwa mogły być zdefiniowane na S, jest, by były określone dla każdego $s_i \in S$.

Uogólniając zatem, w oparciu o definicję przestrzeni probabilistycznej (def. 5.8) i poczynione ustalenia, pożądany w prezentowanym zadaniu planowania PDM można zdefiniować jako czwórkę $<S,A,\xi,\delta>$, gdzie [85,88,100]:

- S – skończony zbiór stanów $S=(s_i)$, w jakich może znaleźć się robot po zrealizowaniu kolejnego kroku elementarnego t_i (współrzędne położenia robota);
- A – skończony zbiór akcji $A=(a_i)$, który definiuje sposoby, jakimi możliwe jest przejście pomiędzy kolejnymi stanami (punktami węzłowymi P_{wi}); A może być zbiorem wszystkich możliwych do zrealizowania przejść z jednego stanu do drugiego zmierzających do osiągnięcia zamierzonego celu;
- ξ – funkcja wartości ocen (5.2);
- δ – funkcja przejścia pomiędzy kolejnymi stanami $\delta_{i+1} = p(s_i,s_{i+1})$ wyrażona prawdopodobieństwem przejścia do wytypowanego stanu s_{i+1}.

5.4. PLANOWANIE ŚCIEŻKI ROBOTA

W opracowanej metodzie, wzorem metody uczenia się ze wzmocnieniem, podstawowym elementem skuteczności jej działania jest odpowiednio zdefiniowana funkcja ocen ξ (5.2). Jej elementami są funkcja $r(P_{wi})$ prawdopodobieństwa zrealizowania zaplanowanego działania i funkcja wag, wartościująca uzyskaną nagrodę. Dodatkowo, funkcja $r(P_{wi})$ jest podstawą przyjętego w metodzie procesu sterowania stochastycznego, opartego na procesach decyzyjnych Markowa. Podstawowym zadaniem jest więc określenie postaci tej funkcji w taki sposób, by spełniała wymagane w zadaniu planowania właściwości PDM (pkt 5.3.1).

5.4.1. Warunki zadania

Próbując wykorzystać PDM do rozwiązania zadania planowania ścieżki robota manipulacyjnego, przyjęto dodatkowe założenie polegające na tym, że środowisko zewnętrzne robota w momencie planowania jest niezmienne w czasie (stacjonarne), co jest warunkiem koniecznym, by proces planowania można było rozpatrywać w kategoriach PDM.

Jeżeli zatem zbiór stanów $S=S(s_i)$ (pkt 5.2.4.2) zależy od środowiska, w którym działa PLANER, to kształtuje on również zbiór podejmowanych przez PLANER akcji $A=A(a_i)$, czyli przejść pomiędzy kolejnymi stanami s_i i s_{i+1} w ramach przyjętej ogólnej strategii działania PLANERA Ψ. Konsekwencją przyjętej stacjonarności środowiska podczas realizacji procesu planowania jest, że zarówno zbiór S, jak i A są skończone, a ich liczność, tzn. $card(S)$ i $card(A)$, jest zawsze dokładnie określona. Wypełnia to podstawowy warunek zachodzenia PDM [68], czyli istnienia na obszarze wykorzystywanej w zadaniu przestrzeni stanów S, przestrzeni probabilistycznej (def. 5.8).

Wykorzystując własność Markowa (5.14) i (5.15) oraz potencjalną konieczność większego uszczegółowienia zadania poprzez dokonanie kolejnych podziałów, liczba nowo powstałych komórek powstających na p-tym poziomie podziału zwiększa się zawsze w postępie 2^3 i wynosi:

$$N^p = \left(2^3\right)^p \tag{5.16}$$

Liczba stanów s_i systemu odpowiada liczebności tworzących go oktantów, czyli:

$$card(S)=N^p$$

Z każdym lokalnym podziałem w miejsce każdego dzielonego rastra powstaje skończona liczba nowych oktantów równa $\Delta N=2^3$ (lokalny podzbiór ΔS^p p stanów, pkt 3.2.2.1.1), w obrębie których następuje wytypowanie kolejnego stanu s_{i+1}^p ze zbioru stanów sąsiednich. Liczność zbioru stanów całego układu $card(S)$ zwiększy się wtedy o pewną wartość ΔN nowych oktantów, czyli:

$$N^p = \sum_p \left[(N^0 - 1) + \Delta N^{p-1} \right] \quad \text{i} \quad \Delta N^p = (2^3)^p \tag{5.17}$$

gdzie: N^0 – liczba rastrów poziomu zerowego (startowego).

Liczność nowo powstałego zbioru ΔS^p jest równa:

$$N' = card(\Delta N^p) = (2^3)^p \tag{5.18}$$

Wynika z tego, że dokonując kolejnych podziałów oktalnych nie zmienia się podstawowego warunku zachodzenia PDM, czyli skończoności zbioru S. Ponieważ zbiór S implikuje liczność zbioru A, zatem spełnienie warunku skończoności zbioru S jest równoznaczne ze skończonością zbioru A, co oznacza, że liczność zbioru A podejmowanych akcji robota (przejść) a_i na danym poziomie podziału p jest również określona i stała. Dokonany podział nie zmienia więc podstawowego warunków skończoności układu.

Każdy podział generuje lokalnie skończony zbiór ΔS^p oktantów – p stanów układu, który stanowi podzbiór zbioru S, czyli: $\Delta S^p \in S$. Opierając się na własnościach prawdopodobieństwa algebry zbiorów wg Kołmogorowa [18,34], można jednoznacznie stwierdzić, że jeżeli dla wybranego oktantu dokonany został podział na poziomie p, wtedy jeżeli

$$S' = \bigcup_i s_i - s$$

oraz

$$\Delta S^p \cup S' = S \text{ oraz } card(S') = card(S - 1) \text{ i } \Delta S^p \text{ oraz } S' \text{ są rozłączne}$$

to

$$p(\Delta S^p \cup S') = p(\Delta S^p) + p(S') \le 1 \quad \text{i} \quad p(S) = 1$$

czyli

$$p(\Delta S^p) + p(S') = \left(p(s_1^p) + p(s_2^p) + \ldots + p(s_8^p) \right) + p(S') \tag{5.19}$$

Tym samym kolejny warunek istnienia przestrzeni probabilistycznej na zbiorze stanów S został spełniony (def. 5.8).

W przypadku robota manipulacyjnego każdy wytypowany przez PLANER stan sąsiedni s_{i+1} musi spełniać podstawowy warunek przynależności do przestrzeni roboczej robota Π_{RP} (pkt 5.2.1), czyli [43,110]:

$$\bigcup_i s_{i+1} \in \Pi_{RP} \quad \text{gdzie} \quad T_Q : q_i \in Q \rightarrow s_i = T_Q(q_i) \in S \tag{5.20}$$

gdzie: q_i – konfiguracja manipulatora określona w układzie stanów wewnętrznych (wektor współrzędnych naturalnych),

Q – przestrzeń konfiguracyjna wewnętrzna manipulatora robota,

T_Q – transformacja prosta kinematyki robota,

s_i, S i Π_{RP} – jak wyżej.

Wynika z tego, że podstawowy warunek realizowalności przemieszczenia wymaga, by wszystkie wartości współrzędnych określających q_i dla poszczególnych osi mieściły się w swoim dopuszczalnym zakresie zmienności i gwarantowały spełnienie warunku (5.20).

5.4.2. Model PLANERA ścieżek bezkolizyjnych

Niech robot znajduje się w pewnym, określonym trzema współrzędnymi kartezjańskimi, punkcie węzłowym P_{wi} wyznaczającym stan s_i na poziomie p, identyfikowanym z początkiem ścieżki. Stan ten nazwano stanem początkowym. Przejście robota (PLANERA) ze stanu początkowego do końcowego s_k będzie się składało wg (5.4) z k kroków. Niech ten ciąg kroków jest realizowany zgodnie z ustaloną, ogólną strategią postępowania PLANERA Ψ. Strategia Ψ rozpięta na A ogranicza zakres swobody działania PLANERA do podzbioru $A^\Psi \in A$. Elementami A^Ψ są wyłącznie te akcje a_i zbioru A, które zostały wytypowane w wyniku zastosowania strategii Ψ. W podobny sposób Ψ określa podzbiór S^Ψ, gdzie:

$$\bigcup_{\Psi: i=1}^{k} s_i = S^\Psi \tag{5.21}$$

Kolejne kroki robota będą się odbywały pomiędzy wskazanymi przez PLANER, zgodnie z Ψ, stanami s_i i s_{i+1}. Ciąg kolejnych zrealizowanych kroków pomiędzy kolejnymi stanami wskazanymi przez PLANER: $s_1 \rightarrow s_2 \rightarrow ... \rightarrow s_{k-1} \rightarrow s_k$ wyznacza planowaną ścieżkę robota *TR*. Jak ustalono (pkt 5.3.1), wytypowanie kolejnego stanu odbywać się będzie z pewnym prawdopodobieństwem, określonym funkcją $p(s_i)$, zaś przejście (podjęta akcja) jest zrealizowane z prawdopodobieństwem równym $p(l,s_i,s_k)$, gdzie l – liczba zrealizowanych kroków t_i planowanej ścieżki. Przyjęto również (pkt 5.2.4.2), że przejścia są możliwe tylko w obrębie zbioru stanów sąsiednich, co oznacza, że każdy kolejny krok t_{i+1} zawsze będzie realizowany wyłącznie do stanu sąsiedniego, czyli dla $l=1$, co zgodnie z metodologią PDM zapisuje się jako $p(1,s_i,s_{i+1}) \equiv p(s_i,s_{i+1})$.

Żeby zatem działania PLANERA mogły być oparte na mechanizmie (5.2), konieczne jest zdefiniowanie funkcji oceny, która będzie stymulować pożądane działania PLANERA, czyli wykonanie zaplanowanej akcji (przejścia) a_{i+1} i osiągnięcie stanu s_{i+1}. Zatem nagrodę, określoną funkcją $r(P_{wi})$, związano z prawdopodobieństwem zrealizowania wytypowanego przez PLANER przejścia ze stanu s_i do s_{i+1} oraz osiągnięcia kolejnego stanu s_{i+1}. Jeżeli każda podjęta akcja zachodzi w jednorodnej i stacjonarnej przestrzeni stochastycznej PDM, w której wszystkie działania są typowe i zachodzą z prawdopodobieństwem $p(s_i)$, wtedy prawdopodobieństwo osiągnięcia celu wynosi [30,68]

$$r_j = f\left(p(s_i), p(s_i, s_j)\right) \tag{5.22}$$

gdzie: $f(\cdot)$ – odpowiednia funkcja wiążąca wartości prawdopodobieństw: $p(s_i)$ i $p(s_i, s_j)$. Oznacza to, że zgodnie z PDM, w kontekście zadania planowania, stan s_j jest osiągalny (w sensie stochastycznym [101,108,147]) ze stanu s_i na przestrzeni 1 kroku. W kontekście zadania planowania oznacza to, że jest on osiągalny, tzn. jest stanem sąsiednim, i dla niego liczba kroków $k=1$.

W ogólnym ujęciu, stan s_j nazywa się osiągalnym ze stanu s_i, jeżeli istnieje liczba l kroków (stanów pośrednich) $l \geq 1$ taka, że $p(l,s_i,s_j)>0$, co oznacza, że pomiędzy nimi zachodzi relacja osiągalności [68], jeżeli tylko spełnione jest dla nich równanie Chapmana-Kołmogorowa [18,34]:

$$p(t+m, s_i, s_j) = \sum_{l \in S} p(t, s_i, s_l) \cdot p(m, s_l, s_j) \tag{5.23}$$

przy warunku, że:

$$s_i \leq s_l \leq s_j \text{ i } s_i, s_l, s_j \in S$$

gdzie: m, t – liczba koniecznych zmian stanów (kroków) prowadzących do zrealizowania przejścia z i do l. Dla zadania planowania oznacza to, że stan końcowy s_k planowanej ścieżki robota jest osiągalny z położenia początkowego s_1, jeżeli osiągalne są kolejno wszystkie wyznaczone przez PLANER punkty węzłowe P_{wi} określające stany pośrednie:

$$s_1 \rightarrow s_2 \rightarrow ... \rightarrow s_i \rightarrow ... \rightarrow s_{k-1} \rightarrow s_k \quad \text{oraz} \quad 0<...<i<...<k$$

Wynika z tego, że całkowita wartość prawdopodobieństwa przejścia ze stanu s_0 do stanu s_k, czyli prawdopodobieństwo zrealizowania planowanej ścieżki robota, przy każdorazowym przejściu równym $k=1$, wynosi:

$$\delta(s_i, s_{i+1}) = p(s_1, s_k) = \sum_{i=1}^{k} p(s_1, s_2) \cdot p(s_2, s_3) \cdot ... \cdot p(s_{k-2}, s_{k-1}) \tag{5.24}$$

Opierając się na określonych własnościach prawdopodobieństwa i zależności (5.23) można stwierdzić, że jeżeli PLANER wybiera kolejny stan s_{i+1} zgodnie z ogólną strategią Ψ, to przejdzie do niego z prawdopodobieństwem równym wg (5.13):

$$p(s_i, s_{i+1}) = \frac{1}{{}_sN}$$

Wybierając zaś stan niezgodny z Ψ zrealizuje go w ogólnym przypadku z prawdopodobieństwem:

$$\bar{p}(s_i, s_{i+1}) = 1 - p(s_i, s_{i+1}) \tag{5.25}$$

rozłożonym na wszystkie pozostałe (nie wybrane) stany sąsiednie, ${}_sN^p$ – zgodnie z (5.13). Relacje, jakie zachodzą dla omawianych zachowań PLANERA, pozwalają zdefiniować tzw. *macierz **M** funkcji przejść* PDM dla realizowanego zadania w postaci (5.26).

$$M(k,s_i,s_j)=\begin{array}{|c|c|c|c|c|c|c|}\hline i \\ j & 1 & 2 & 3 & 4 & \ldots & k \\ \hline 1 & p(s_1) & p(s_1,s_2) & 0 & 0 & \ldots & 0 \\ 2 & p(s_2,s_3) & p(s_2) & p(s_2,s_3) & & \ldots & 0 \\ 3 & & p(s_3,s_4) & p(s_3) & p(s_3,s_4) & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ k & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline \end{array} \quad (5.26)$$

Przedstawiona macierz funkcji przejść M (5.26) jest macierzą stochastyczną [68] spełniającą warunek, że $\sum_{j=1}^{k} p(s_i,s_j)=1$ dla każdego j, co oznacza, że może być uważana za reprezentację algebraiczną omawianego PDM.

Stan s_k – zgodnie z algorytmem uczenia się ze wzmocnieniem – jest tzw. stanem pochłaniającym [30,68], tzn. takim, dla którego $p(s_k,s_k)=1$. Oznacza to, że PLANER realizujący swoje działania w oparciu o PDM nie może już z niego wyjść. Stan pochłaniający zamyka więc PDM i kończy działanie PLANERA.

Zgodnie ze zdefiniowanym PDM (5.26) pozostanie w bieżącej komórce jest możliwe tylko dla stanu pochłaniającego, a taki jest stan końca planowanej ścieżki, dla którego $p(s_k)=1$ i $p(s_k,s_k)=1$. Oznacza to, że dla każdego s_j, dla którego $j \le k$ i $p(s_j,s_j) \le 1$, pozostanie w nim nie jest dopuszczalne.

Macierz funkcji przejść $M(k,s_i,s_j)$ pokazuje sposób przejścia PLANERA od stanu s_0 do s_k na przestrzeni k kroków. Stochastyczna macierz przejść M, dla której spełnione jest równanie Chapmana-Kołmogorowa (5.23), nazywamy jednorodną macierzą funkcji przejść i taką macierzą jest M (5.26).

Tym samym omawiany proces, dzięki warunkowi stałości zbioru stanów S i akcji A, spełnia pożądane warunki stacjonarnego (definicja 5.7) i jednorodnego (definicja 5.6) procesu decyzyjnego Markowa.

5.4.3. Ocena podjętych działań

Posługując się zdefiniowanym dla zadania planowania bezkolizyjnych ścieżek robotów PDM (5.26), można stwierdzić, że działania PLANERA zakończą się pełnym sukcesem w przypadku, gdy osiągnie on stan s_{i+1} wynikający z przyjętej strategii Ψ i na tej podstawie będzie mógł wytypować z prawdopodobieństwem $p(s_i,s_{i+1})$ kolejną akcję, dzięki której osiągnie kolejny stan s_{i+1}, również zgodny z Ψ. Ponieważ stan bieżący określony $p(s_i)$

decyduje o możliwościach zrealizowania przejścia (s_i,s_{i+1}), więc wartość osiągniętego przez PLANER sukcesu zdobytego w jednym kroku określa zdobytą oceną, która jest równa wg (5.22) i (5.26):

$$r_{i+1} = p(s_i)\cdot p(s_i,s_{i+1}) \quad (5.27)$$

i ogólnie dla k kroków [68]:

$$r_k = \sum_k p(s_1)\cdot p(s_1,s_2)\cdot p(s_2,s_3)\cdot \ldots \cdot p(s_i,s_{i+1})\cdot \ldots \cdot p(s_{k-1},s_k)$$

Tak wyznaczona wartość oceny działań PLANERA ustalona jest dla określonego stanu systemu, w którym zbiory S i A są jednoznacznie określone i dla którego zachodzi (5.20). Jednakże, lokalnie, ze względu na konieczność uszczegółowienia analizy, konieczne jest dokonywanie kolejnych podziałów. Opierając się na ergodyczności PDM, czyli wykorzystując ich własność mówiącą o tym, że w systemie, którego modelem jest PDM, wszystkie reakcje są typowe [18,30], uogólniono wartość zdobywanych ocen do postaci:

$$r_{i+1} = \gamma_{i+1}\cdot p(s_i)\cdot p(s_i,s_{i+1}) \quad (5.28)$$

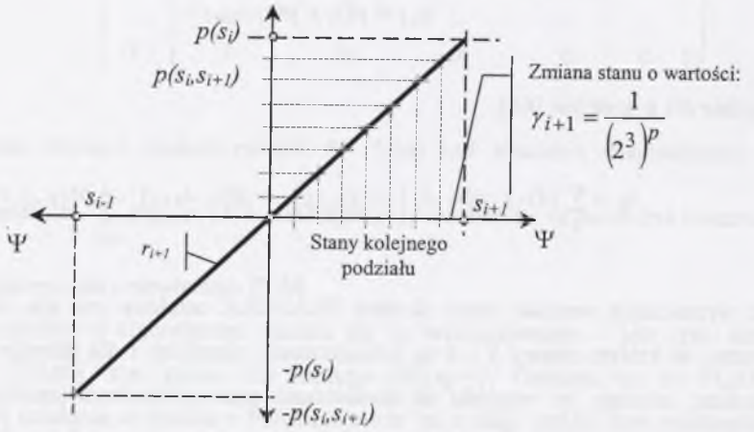
gdzie: γ_{i+1} – przyjęty współczynnik proporcjonalności, $\gamma = (0,1]$. Przy czym $\gamma = 1$ dla podziału podstawowego i $\gamma < 1$ proporcjonalnie dla podziałów $p > 0$. γ wyznacza zatem zależność pomiędzy prawdopodobieństwem zajścia stanu s_i na poziomie zerowym i p-tym. Tym samym funkcję oceny r dla określonego PDM (5.26) ustalono w postaci:

$$r_{i+1} = \begin{cases} \gamma_{i+1}\cdot\left(p(s_i)\cdot p(s_i,s_{i+1})\right), & gdy \quad s_{i+1}\in\Psi \\ 0, & gdy \quad i=0,\ p(s_i,s_i)=0 \\ -\gamma_{i+1}\cdot\left(p(s_i)\cdot p(s_i,s_{i+1})\right), & gdy \quad s_{i+1}\notin\Psi \end{cases} \quad (5.29)$$

Współczynnik proporcjonalności γ określony dla p w postaci:

$$\gamma_{i+1} = \frac{1}{\left(2^3\right)^p} > 0, \quad p \quad 0 \quad (5.30)$$

wynika z konieczności dokonania podziałów wytypowanej komórki (rastra) jako stanu s_{i+1}. Dla tych warunków przebieg funkcji r_{i+1} pokazano na rys. 5.10.



Rys. 5.10. Wpływ parametru γ na wartości funkcji $p(s_i)$ i $p(s_i,s_{i+1})$

Fig. 5.10. Impact of parameter γ on the value of function $p(s_i)$ and $p(s_i,s_{i+1})$

Podstawowym warunkiem, jaki muszą spełnić funkcje rozkładu prawdopodobieństwa $p(s_i)$ i $p(s_{i+1})$ wykorzystane w PDM (pkt 5.3.1), jest, by były one określone dla każdego stanu $s_i \in S$. Funkcja współczynnika γ (5.30) spełnia ten warunek.

Warunek (5.28) nie zmienia warunków stacjonarności omawianego PDM. Funkcja przejścia $\delta(s_i,s_{i+1})$, uwzględniająca współczynnik γ, jedynie kwalifikuje PDM do klasy procesów stacjonarnych o przyrostach niezależnych od czasu [101].

5.4.3.1. Funkcja prawdopodobieństwa osiągnięcia stanu s_i

Dokonany pierwotnie podział (startowy) zewnętrznej przestrzeni zadania pozwala na wyznaczenie prawdopodobieństwa osiągnięcia wybranego stanu s_i. Jeżeli parametrem tego podziału przestrzeni robota Π_{SC} jest bok b^s rastra ($p=0$, pkt 5.2.4.2), wtedy liczba rastrów N^s, na jaką można podzielić Π_{SC} jest równa:

$$N^s = \frac{\Pi_{SC}}{\left(b^s\right)^3} \tag{5.31}$$

Liczba rastrów należąca do przestrzeni bezkolizyjnej Π_w (pkt 5.2.4.2), wyznaczona z dokładnością wystarczającą do przeprowadzenia niniejszego rozumowania, jest równa dla $p=0$:

$$N_w = \mathrm{mod}\left(\frac{\Pi_w}{\left(b^s\right)^3}\right) \tag{5.32}$$

Ogólnie, w oparciu o (5.13) można stwierdzić, że na poziomie p liczba rastrów należących do przestrzeni bezkolizyjnej jest równa:

$$N_w^p = (N_w - 1) + \sum_p (2^{3p} - 1) \tag{5.33}$$

Gdy oprzemy się na klasycznej definicji prawdopodobieństwa [18,34,108,147] odniesionej do zagadnień geometrycznych [18], wybranie dowolnego rastra pustego na poziomie startowym zachodzi na podstawie (5.32) z prawdopodobieństwem:

$$p({}_s s_w^p) = \frac{1}{N_w^p} = \frac{1}{(N_w - 1) + \sum_p (2^{3p} - 1)} \tag{5.34}$$

Funkcja prawdopodobieństwa osiągnięcia stanu pustego (5.34) zgodnie z ogólną strategią Ψ działania PLANERA jest kolejnym elementem spełniającym warunek istnienia przestrzeni probabilistycznej (def. 5.8) PDM, w której działa PLANER.

5.4.3.2. Prawdopodobieństwo przejścia

Konstruując funkcję prawdopodobieństwa $p(s_i,s_{i+1})$ przejścia PLANERA ze stanu bieżącego s_i do kolejnego s_{i+1} na przestrzeni jednego kroku $t_{i+1}=1$, należy jednoznacznie stwierdzić, że ze względu na ewentualne podziały rastrów, zachodzące na wyższych poziomach podziału, kolejny krok PLANERA, w granicach zbioru stanów sąsiednich, teoretycznie może się odbyć zarówno do stanu, który istnieje na poziomie zerowym, jak i do takiego, który powstał lokalnie na poziomie $p>0$. Jednocześnie, ze względu na ustaloną funkcję oceny (5.2), pożądane jest, by:

- planowana ścieżka była zrealizowana przy możliwe najmniejszej liczbie kroków (5.4), a zatem, by w kroku t_{i+1} prowadzącym do osiągnięcia stanu s_{i+1} była możliwie najdłuższa, tym samym zapewniała maksymalne zbliżenie się do celu,
- przejście oznaczało wykorzystanie najlepszych szans na ociągnięcie wskazanego celu, wyrażonych możliwie największą wartością prawdopodobieństwa jego osiągnięcia, a przez to największą uzyskiwaną oceną w tym kroku.

Aby sprostać tym wymaganiom, określono wartość prawdopodobieństwa przejścia, pamiętając, że przejście to powinno wskazać kolejny punkt węzłowy poszukiwanej ścieżki robota, a zatem jej argumentem powinno być jego położenie w stanie s_{i+1}. Planowane przejście PLANERA jest zgodne z wektorem przejścia $\vec{d}_{i+1}$ (def. 5.1 i 5.2).

Dla takich warunków, z każdym przejściem zgodnym z wektorem przejścia $\vec{d}_{i+1}$, związana jest kolejna akcja a_{i+1}, która zrealizowana w kroku t_{i+1} na drodze o długości $|\vec{d}_{i+1}|$ pozwala osiągnąć kolejny stan s_{i+1}, a przez to kolejny punkt węzłowy P_{wi+1}.

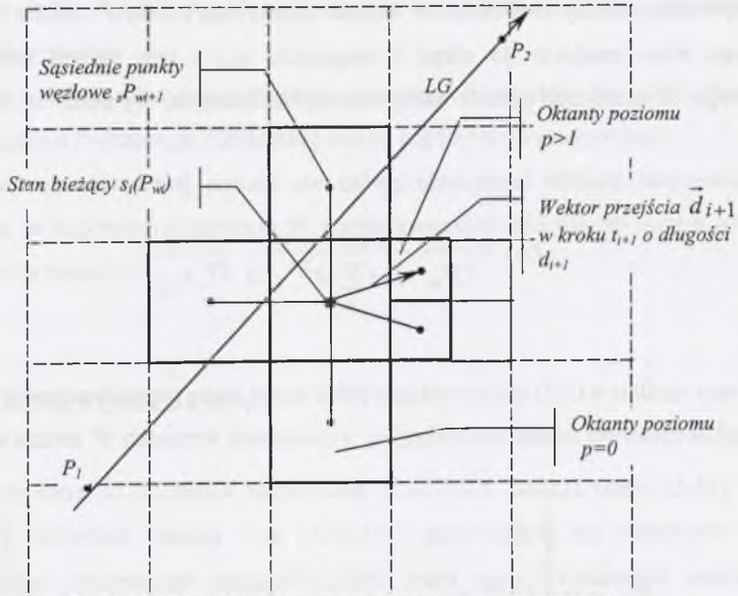
Ustalono zatem wartość pradopodobieństwa zrealizowania, na przestrzeni jednego kroku t_{i+1} o długości równej d_{i+1}, przejścia pomiędzy dwoma sąsiednimi stanami, jako:

$$\delta(s_i, s_{i+1}) = p(s_i, s_{i+1}) = \frac{d_{i+1}}{D_{s P_{wi+1}}} \tag{5.35}$$

gdzie: d_{i+1} - długość drogi do wytypowanego stanu s_{i+1} (rys. 5.11), przy czym:

$$d_{i+1} = \begin{cases} b^s & \text{dla} \quad p = 0 \\ \dfrac{\left[\left(b^s + b^p_{i+1}\right)^2 + \left(b^p_{i+1}\right)^2\right]^{\frac{1}{2}}}{2} & \text{dla} \quad p > 0 \end{cases} \tag{5.36}$$

$D_{s P w_{i+1}}$ - suma długości wszystkich dróg $|{}_s\vec{d}_{i+1}|$ prowadzących do wszystkich stanów sąsiednich ${}_s s_{i+1}$.



Rys. 5.11. Układ zbioru stanów sąsiednich (rzut na płaszczyznę)

Fig. 5.11. The arrangement of the set of contiguity states (plain view)

Tak określone prawdopodobieństwo przejścia odpowiada podjętej akcji PLANERA (def. 5.2) w kroku t_{i+1}, na drodze d_{i+1}.

Z przeprowadzonego rozumowania wynika, że ustalona jak w (5.35) wartość prawdopodobieństwa przejścia robota ze stanu bieżącego s_i do kolejnego, wyznaczonego przez PLANER s_{i+1}, zależy wprost od odległości związanego z nim punktu węzłowego P_{wi+1}. Zatem PLANER powinien preferować punkty węzłowe określające kolejne stany należące do zbioru stanów sąsiednich ${}_s s_{i+1}$, które będą możliwie najdalej oddalone od położenia bieżącego, tj. dla których $|\vec{d}_{i+1}| \to \max(d_i)$, gdyż dla nich wartość zdobywanej oceny będzie największa. Tym samym uzyskano wartość prawdopodobieństwa przejścia wyrażoną w funkcji długości drogi do kolejnego stanu s_{i+1} i w sposób zupełny wypełniono wymogi wynikające z definicji przestrzeni probabilistycznej (def. 5.8)

5.4.3.3. Ocena podjętych działań

Reasumując, można dzięki określonym prawdopodobieństwom:

- osiągnięcia wytypowanego stanu (5.34),
- przejścia pomiędzy stanami sąsiednimi (5.35)

w sposób jednoznaczny stwierdzić, że wartość oceny, jaką uzyska PLANER za wyznaczenie kolejnego stanu przejścia do niego i osiągnięcia go, a tym samym kolejnego punktu węzłowego określającego poszukiwaną ścieżkę bezkolizyjną, wyniesie wg (5.29), (5.34) i (5.35):

$$r_{i+1} = \frac{\gamma_{i+1}}{(N_w - 1) + \sum_{p}(2^{3p} - 1)} \cdot \frac{d_{i+1}}{D_s P_{w_{i+1}}} \tag{5.37}$$

Jeżeli teraz zgodnie z (5.2) dobrze radzący sobie uczeń może uzyskać najwyższe oceny, to dla potencjalnie możliwej ścieżki bezkolizyjnej wyznaczonej w ramach Ψ można zapisać:

$$\xi = \max_{i \to \min} \left[E \left(\sum_{i=1}^{k} \omega_i \cdot \frac{\gamma_{i+1}}{\left[(N_w - 1) + \sum_{p}(2^{3p} - 1) \right] \cdot D_s P_{w_{i+1}}} \cdot d_{i+1} \right) \right] \tag{5.38}$$

Zależność ta, jak określono w pkt. 5.1.2 i 5.2.2, może być uznana za proste kryterium optymalizacyjne poszukiwanej ścieżki mówiące, że najlepsza jest ta spośród znalezionych, która umożliwia uzyskanie najwyższej oceny przy najmniejszej liczbie kroków PLANERA. Równocześnie kryterium to powoduje, że PLANER w warunkach, gdy nie jest zdeterminowany sytuacjami przymusowymi, np. koniecznością określoną położeniem końca planowanej ścieżki s_k, będzie wybierał – ze względu na (5.30) i (5.37) – takie rozwiązania (stany sąsiednie), które będą na możliwie najniższym poziomie podziału, czyli będzie się starał wybierać te rastry, które dla $p=0$ będą określone jako „puste".

Jednocześnie zależność (5.38) odpowiada definicji (def. 5.3) zadania robota (5.4). Tym samym można stwierdzić wg (5.4) i (5.38), że:

$$\xi = TR$$

Na podstawie przeprowadzonej analizy można wysnuć następujące wnioski:

- jest możliwe zdefiniowanie przestrzeni stochastycznej na bazie zewnętrznej przestrzeni roboczej robota, a tym samym zdefiniowanie PDM, dzięki któremu udaje się wypracować system oceniania PLANERA poruszającego się w przestrzeni zadania i rozwiązującego problem planowania,
- dokonywanie lokalnych uszczegółowień przestrzeni (podziałów dla $p>0$) nie zmienia jakości i rodzaju tworzonego PDM (dalej jest on ergodyczny i stacjonarny),
- metoda w zaprezentowanej postaci nie zabezpiecza przed efektem błądzenia; należy oczekiwać, że odpowiednia strategia Ψ i zróżnicowany dzięki wagom system ocen może ograniczyć to zjawisko.

5.5. DYSKRETNA FUNKCJA WAG

Wykorzystywana do oceniania zachowania PLANERA funkcja oceny (5.38) wymaga zdefiniowania dyskretnej funkcji wag [3,30,163] pozwalającej na określenie wartości ω_{i+1} dla każdego typowanego przez PLANER stanu s_{i+1}. Wyznaczenie wartości r_{i+1} dokonane zostało, przy założeniu że wszystkie stany analizowanego systemu są jednakowo wartościowe, a różnice wynikają jedynie z warunków wyboru kolejnego stanu s_i zgodnie z przyjętą strategią Ψ lub niezgodnie z nią. Jednak w ogólnym przypadku atrakcyjność poszczególnych, potencjalnie możliwych do wyboru stanów jest różna, zależnie od miejsca ich położenia w analizowanej przestrzeni. Atrakcyjność tę może obniżyć:

- konieczność dokonania podziałów kształtująca parametr γ (5.30),
- bliskość obiektów sceny, mogących spowodować kolizję robota.

Druga z tych przyczyn, kształtująca dyskretną funkcję wag, wymaga szczegółowego zdefiniowania.

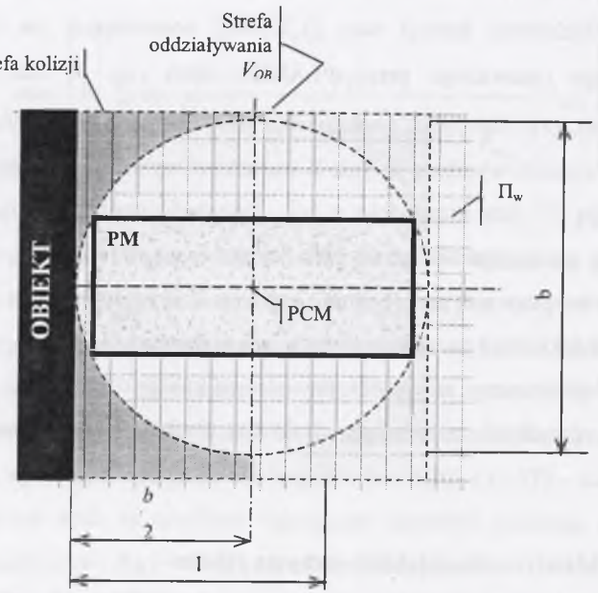
5.5.1. Strefa oddziaływania obiektów sceny na robota

Założono, że kolizja robota z obiektami sceny zachodzi wtedy, gdy występuje jeden z przypadków, kiedy punkt węzłowy planowanej ścieżki robota został wyznaczony:

- w obszarze przestrzeni zajętej przez jeden z jej obiektów,
- w odległości mniejszej od przyjętej umownie granicy, po przekroczeniu której robot, lub przedmiot przez niego manipulowany, jest w kontakcie z obiektem sceny. Oznacza to, że przedmiot lub robot ma z obiektem sceny przynajmniej jeden punkt wspólny.

Warunki te określa zależność (5.7).

Ze względu na możliwą kolizję robota z obiektami OB^j sceny związano z każdym z nich pewną strefę „negatywnego" oddziaływania V_{OB}, która ze względu na bezpośrednie sąsiedztwo robota i obiektu, a co za tym idzie – możliwości wystąpienia kolizji, będzie wymuszać reakcję „odpychania" robota. Oddziaływanie to może wpływać na wybór bardziej bezpiecznego miejsca. Wybór ten może być stymulowany poprzez obniżenie „atrakcyjności" obszaru przestrzeni sceny mogącego zostać wytypowanym do zrealizowania kolejnego kroku t_i. Obniżenie atrakcyjności typowanego stanu może zajść poprzez zmianę wagi, przez co zmianie ulegnie wartość uzyskiwanej oceny wraz ze zbliżaniem się robota do obiektu OB^j. Ustalono, że obszar V_{OB}, jakim oddziałuje każdy obiekt OB^j sceny robota na jego przestrzeń bezkolizyjną Π_w w jego bezpośrednim sąsiedztwie, określa wartość wagi C_1 i wynosi (rys. 5.12) [92,93]:



Rys. 5.12. Wyznaczanie V_{OB}. PM – przedmiot manipulowany, PCM – punkt charakterystyczny manipulacji

Fig. 5.12. Designation of V_{OB}. PM – manipulated object; PCM – manipulation characteristic point

$$C_1 = \begin{cases} 0 & dla\ l_1 = 0 \\ 1 - \dfrac{1}{l_1} & dla\ l_1 \in \left(\dfrac{b}{2}, b\right) \\ 1 & dla\ l_1 \geq b \end{cases} \qquad (5.39)$$

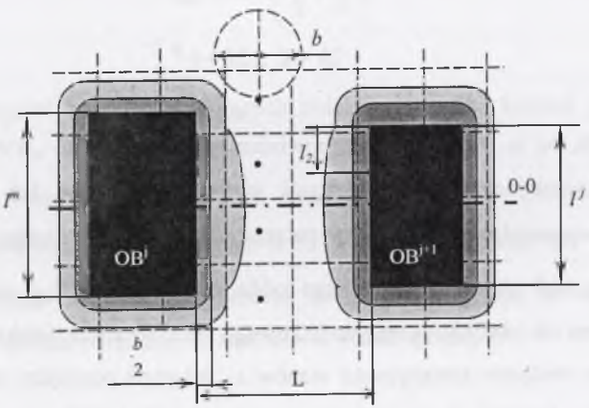
gdzie: l_1 – rzeczywista odległość robota (programowanego punktu charakterystycznego robota) od przeszkody, b – wartość określona wg (5.8).

By możliwe było swobodne kształtowanie strefy oddziaływania V_{OB}, np. różnego dla różnych obiektów sceny, proponuje się wprowadzić do zależności (5.39) pewien dodatkowy parametr, tzw. współczynnik korekcyjny C_1^j. Przyjmuje się, że ze względów bezpieczeństwa C_1^j może przybierać wartość nie mniejszą niż 1. Jednakże nieuzasadnione zawyżanie wartości C_1^j może spowodować, że w przypadku stosunkowo blisko leżących obiektów przestrzeń pomiędzy nimi nie będzie możliwa do wykorzystania. Dlatego proponuje się, że przybiera on wartości z przedziału $C_1^j = [1,2]$. Wtedy wartość wagi C_{OB^j} kształtującej ocenę w obszarze wokół obiektu OB^j przybierze postać:

$$C_{OB^j} = C_1 \cdot C_1^j \qquad (5.40)$$

Należy zwrócić uwagę, że tak wyznaczona wartość wagi C_1 umożliwia penetrację przestrzeni wokół obiektu na odległość nie mniejszą niż $\frac{b}{2}$.

Ze względu na rzeczywiste warunki, jakie mogą zachodzić w czasie pracy robota, w przestrzeni roboczej rozpatrzono trzy przypadki charakterystyczne, związane z oddziaływaniem strefy V_{OB} i zajścia kolizji (rys. 5.13) [15,92,93,133,136].



Rys. 5.13. Sposób wyznaczania parametru C_2 [93]

Fig. 5.13. Mode of designating parameter C_2 [93]

Przypadek 1

Zachodzi wtedy, gdy spełniony jest warunek $L >> b$. Wtedy reakcja na sąsiedztwo kilku obiektów sceny (przeszkód) nie implikuje innych warunków niż typowe, zachodzące w modelowanym otoczeniu robota. Zatem obowiązują wszystkie poczynione do tej pory ustalenia, a wartość wagi $C_1^j = 1$.

Przypadek 2

Zachodzi w przypadku, gdy: $b > b^n > L$. Wtedy nie ma możliwości przeprowadzenia robota w przestrzeni powstałej pomiędzy blisko sąsiadującymi ze sobą obiektami OB^j i $C_1^j = 0$.

Przypadek 3

Przypadek ten ma miejsce wtedy, kiedy $L > b$ i nie zachodzi *przypadek 1*, tzn. gdy odległość pomiędzy obiektami jest „niewiele" większa niż wartość b. Oznacza to, że robot musi przemieszczać się pomiędzy obiektami sceny wyjątkowo ostrożnie, gdyż jego ruch odbywa się w bezpośredniej ich bliskości. Przyjęto, że bezpośrednia bliskość robota i obiektu sceny zachodzi wtedy, kiedy nie jest możliwe wybranie, jako kolejnego s_{i+1}, rastra sąsiadującego obok, a możliwy jest tylko ruch do przodu lub do tyłu. Warunek ten można określić zależnością (rys. 5.13):

$$2b < L \leq 2b + b^p \tag{5.41}$$

Ze względu na to, że obszar taki powinien być możliwy do „wykorzystania" tylko w wyjątkowych sytuacjach (np. konieczność obsługi obrabiarki, lub gdy nie ma innej możliwości), wprowadzono dodatkowy parametr C_2 korygujący wagę C_{OB^j}, powodujący „wykształcenie się" obszaru wzajemnego oddziaływania stref V_{OB} sąsiadujących ze sobą obiektów. Celem tak zdefiniowanego dodatkowego oddziaływania obiektu jest [92,93]:

- dodatkowe obniżenie atrakcyjności stanów s_i leżących pomiędzy obiektami, ale w taki sposób, by korekta wagi jedynie zniechęciła PLANER robota do swobodnego penetrowania tego obszaru,
- umożliwienie wejścia w ten obszar na zasadzie „im głębiej, tym trudniej", co pozwoliłoby na skorzystanie z niego jedynie wtedy, gdy jest to konieczne ze względów

technologicznych, oraz ułatwiło opuszczenie tej strefy poprzez nagradzanie kroków zmierzających do jej opuszczenia.

Dlatego proponuje się ustalić wartość parametru C_2 następująco:

$$C_2 = \begin{cases} 1 & dla\ \ k = 0 \\ \dfrac{1}{l_2} & dla\ k \in \left(0, \dfrac{l_{OB}}{2}\right\rangle \\ l_2 - \dfrac{1}{l_2} & dla\ k \in \left(\dfrac{l_{OB}}{2}, l_{OB}\right) \end{cases} \tag{5.42}$$

gdzie: l_2 – głębokość, na jakiej PLANER wytypował kolejny stan, l_{OB} – długość boku przeszkody, wzdłuż którego realizowane jest przejście robota.

Przyjęta postać współczynnika korekcyjnego C_2 jest jednak obarczona pewną wadą. Mianowicie, elementem decydującym jest długość boku, wzdłuż którego obiekty sąsiadują ze sobą. Ale ze względu na to, że celem przyjętej korekcji jest zarówno zniechęcenie robota do penetrowania tego obszaru, jak i zachęcenie go do jego opuszczenia, przyjęto, że ten parametr korekcyjny narasta do połowy długości boku przeszkody, czyli spełnia warunek, że:

$$l_{OB} = \min\left(\frac{l^j}{2}, \frac{l^m}{2}\right)$$

gdzie: l^j, l^m – długości boków sąsiadujących obiektów, wzdłuż których planowane jest przejście robota.

Przyjmuje się również, że w sytuacji wzajemnej bliskości przeszkód określonej *przypadkiem 3* oddziaływania określone wagami C_2 i C_{OB^j} wzmacniają się.

Zatem, uogólniając, omówione dotychczas przypadki wpływają na wartość wagi ω korygującej przyjęty system oceniania działań robota w taki sposób, że:

$$\omega_i = (C_1 \cdot C_1^j \cdot C_2)_i \tag{5.43}$$

Tak więc, wykorzystując określony system wag, można bezpośrednio wpływać na sposób nagradzania PLANERA próbującego w możliwie najlepszy sposób, czyli taki, który zagwarantuje najwyższą ocenę (5.38), wyznaczyć bezpieczną (bezkolizyjną) ścieżkę do wskazanego celu w złożonym środowisku technologicznym robota.

Mimo że ruch PLANERA w podzielonej oktalnie przestrzeni odbywa się skokowo (wg t_i), stosownie do określonego gradientu planowania, to zasadność określania wartości l i sensu ustalonych wag nie zmienia się.

5.6. STRATEGIA POSZUKIWANIA BEZKOLIZYJNYCH ŚCIEŻEK ROBOTA

Omawiając do tej pory działania projektowanego PLANERA bezkolizyjnych ścieżek robota, wielokrotnie powoływano się na fakt istnienia ogólnej strategii Ψ działania PLANERA, która umożliwia mu rozwiązywanie postawionego zadania. Strategia Ψ, podobnie jak w metodzie uczenia się ze wzmocnieniem, opiera się na długoterminowej maksymalizacji zdobywanych nagród, co wyrażone jest zależnością (5.38). Jednak ze względu na specyfikę omawianego problemu, wymaga ona zaadaptowania do zadania planowania. Podobnie jak w metodzie uczenia się ze wzmocnieniem, ogólna strategia wartościująca wymaga zastosowania szczegółowych strategii (lokalnych) realizacji zadania (pkt 5.1.1). Strategia Ψ jest niezależna od strategii lokalnych, które można wprowadzić do algorytmu działania PLANERA w zależności od specyfiki rozwiązywanego zadania [30,97]. Jednym z celów wprowadzenia do działań PLANERA bezkolizyjnych ścieżek robota, tzw. strategii lokalnych, może być ograniczenie procesu błądzenia przypadkowego, a tym samym ograniczenie wielkości uzyskiwanego w procesie planowania drzewa decyzyjnego V (pkt 3.2.2.1 i 5.2.4.2).

Omawiana strategia lokalna ψ rozumiana jest jako sposób racjonalnego postępowania PLANERA reprezentującego poruszający się robot w środowisku sceny, prowadzący do osiągnięcia zamierzonego celu możliwie najkrótszą drogą [46,70,97,98,119,127,133,165]. Jej podstawowym celem jest ograniczenie procesu błądzenia przez zawężenie obszaru poszukiwań rozwiązań dopuszczalnych. Tym samym osiągnięty zostanie sprzyjający zadaniu warunek ograniczenia przestrzeni poszukiwań i przyspieszony proces rozwiązania postawionego zadania w złożonym geometrycznie środowisku technologicznym. Opiera się ona zatem na dwóch podstawowych celach, jakimi są:

- ograniczenie w możliwie najszerszym zakresie procesu błądzenia, jednak bez szkody dla ogólności uzyskiwanych rozwiązań, tak by zastosowany do przeszukiwania powstałego drzewa decyzyjnego algorytm heurystyczny był maksymalnie efektywny i szybki,
- umożliwienie wykrycia przeszkód i wyznaczenie najkrótszej ścieżki w przypadku ich omijania, tak by zrealizować tym samym proste kryterium optymalnościowe.

Przy rozwiązywaniu tych dwóch zadań strategia Ψ powinna umożliwić znalezienie rozwiązania obiektywnie najlepszego (najkrótsza ścieżka) w danych warunkach geometrycznych sceny, które pozbawione by było uznaniowego charakteru programisty robota.

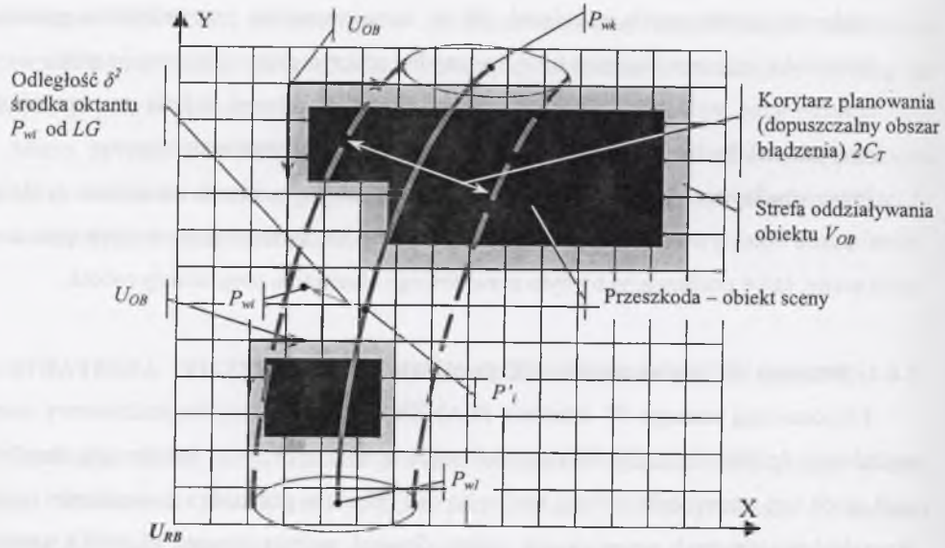
5.6.1. Strategia Ψ wyznaczania najkrótszej ścieżki robota

Proponowana strategia Ψ działania PLANERA wykorzystuje jako podstawowy warunek najkrótszej ścieżki, łączącej bieżące położenie robota $s_1(P_{w1})$ z położeniem docelowym $s_k(P_{wk})$, co jest szczególnie ważne i trudne do osiągnięcia w przypadku konieczności omijania przeszkód znajdujących się na drodze robota. Stopień spełnienia przez PLANER warunków określonych omawianą strategią wyznacza wartość wagi C_3 [97,98] modyfikującej ocenę, jaką PLANER-uczeń może uzyskać w wyniku zrealizowania wytypowanego w oparciu o PDM kroku.

Strategia rozpoczyna swoje działanie od połączenia dwóch danych położeń: początkowego, identyfikowanego z położeniem bieżącym P_{w1} i końcowego P_{wk} planowanej ścieżki TR linią geodezyjną LG [18,34,43], co wynika ze sferycznego układu kinematycznego robota (pkt 5.2.1). Zatem: $LG(P_{w1},P_{wk})$, (rys. 5.14) [97,98]. Linia LG (w układzie płaszczyzny jest prostą [18,34]) wyznacza główny kierunek poszukiwań od $P_1=P_{w1}$ do $P_2=P_{w2}$, a zadaniem PLANERA jest typowanie takich stanów pośrednich $s_i(P_{wi})$, które spełniają postawiony warunek najmniejszej odległości od LG (w sensie normy euklidesowej).

Działanie strategii Ψ uzależnione jest od pojawienia się przeszkody na określonym linią geodezyjną, głównym kierunku poszukiwań ścieżki robota. Ogólny algorytm działania strategii Ψ jest następujący:

1) dla zadanego położenia końca planowanej ścieżki robota określić linię geodezyjną LG i sprawdzić, czy przecina ona którąkolwiek z przeszkód OB^j znajdujących się w przestrzeni Π_{RB} otaczającej robot,
2) jeżeli na głównym kierunku poszukiwań ścieżki robota nie znajduje się żadna z przeszkód, czyli gdy $\forall_j (LG \cap OB^j) = 0$, to PLANER stosuje strategię ψ_1,



Rys. 5.14. Plan strategii Ψ [97,98]

Fig. 5.14. Plan of strategy Ψ [97,98]

3) jeżeli na głównym kierunku poszukiwań ścieżki robota znajduje się przeszkoda, tzn. gdy zachodzi warunek, że $\forall_j(LG \cap OB^j) > 0$, to PLANER stosuje strategię ψ_2.

Zaproponowana strategia Ψ rozwiązywania postawionego zadania składa się zatem z dwóch lokalnych strategii pomocniczych ψ_1 i ψ_2.

5.6.1.1. Strategia pomocnicza ψ_1 „prosto do celu"

Strategia ψ_1 „prosto do celu" [97,98] wymusza na PLANERZE, by każdy kolejny krok t_i planowanej ścieżki prowadził do wytypowania stanu s_i (przeszukiwanie drzewa decyzyjnego) pokrywającego się lub umiejscowionego jak najbliżej LG. Kryterium poprawności działań PLANERA w oparciu o ψ_1 jest parametr C_T (rys. 5.14). Parametr ten przyjęto jako zmienny, zależny do warunków występujących w omawianym środowisku i od konkretnego zadania. Wyznacza on tzw. korytarz planowania, w obrębie którego PLANER, w ramach strategii ψ_1, może działać w oparciu o mechanizm błądzenia przypadkowego i nie jest to karane zaniżaniem zdobytej oceny. Korytarz ten, o średnicy $2C_T$, określono w celu ograniczenia przestrzeni poszukiwań. Parametr C_T charakteryzuje się następującymi własnościami:

- dobierany jest każdorazowy dla realizowanego zadania,
- określa maksymalną, dopuszczalną odległość od LG, która nie wpływa na wartość wagi C_3 obciążającej zachowanie niezgodne z obowiązującą strategią ψ_1.

Przyjęto, że wartość C_T nie powinna być mniejsza od boku rastra powstałego na poziomie startowym podziału przestrzeni ($p=0$) [97,98].

Zgodnie z ψ_1, każde zejście PLANERA z toru wyznaczonego przez korytarz planowania LG i C_T jest „karane" odpowiednio niższą oceną przez wyznaczenie jej wagi C_3 *1* [97,98]. Przyjęto, że wartość wagi C_3 zależy proporcjonalnie od odległości, na jaką PLANER oddala się od LG. W przypadku dążenia PLANERA do zdobywania zawsze oceny najwyższej z możliwych mechanizm ten powoduje odstąpienie od podążania wzdłuż LG tylko w przypadkach wymuszonych koniecznością omijania przeszkód (konieczność zastosowania strategii ψ_2). Kryterium wyboru stanu s_{i+1} określającego następny krok planowanej ścieżki jest zatem zachowanie minimalnej odległość δ^2 od prostej LG wyrażonej zależnością [18,34]:

$$\delta^2 = \frac{[(X_i - X_i')v - (Y_i - Y_i')u]^2}{u^2+v^2+w^2} + \frac{[(Y_i - Y_i')w - (Z_i - Z_i')v]^2}{u^2+v^2+w^2} + \frac{[(Z_i - Z_i')u - (X_i - X_i')w]^2}{u^2+v^2+w^2} \tag{5.44}$$

gdzie: $P_i(x_i,y_i,z_i)$ – punkt węzłowy planowanej ścieżki wskazany przez PLANER w oparciu o ψ_1, $P'_i(x'_i,y'_i,z'_i)$ – rzut P_p na LG, $u=[x_i - x_i']$; $v=[y_i - y_i']$; $w=[z_i - z_i']$.

Spełniając omówione warunki proponuje się, by waga C_3 była określona wg reguły:

$$C_3 = \begin{cases} 1 & gdy \quad \delta^2 \le C_T \\ \dfrac{1}{\delta^2} & gdy \quad \delta^2 > C_T \end{cases} \tag{5.45}$$

Na bazie strategii ψ_1 PLANER buduje zawężoną przestrzeń poszukiwań, dla której zakres przeszukiwania otrzymanego drzewa-grafu jest znacznie mniejszy, gdyż efekt błądzenia ograniczony jest korytarzem poszukiwań określonym parametrem C_T.

5.6.1.2. Strategia pomocnicza ψ_2 „omiń"

Jest to strategia stosowana doraźnie w sytuacji wystąpienia konieczności ominięcia przeszkody OB^j znajdującej się na planowanej ścieżce robota. Strategia korzysta z metody grafu widoczności [43,106] pozwalającej na wyznaczenie kierunku obejścia napotkanej przeszkody. Metoda grafu widoczności jest jedną z najbardziej skutecznych i sprawnych metod planowania bezkolizyjnych trajektorii robotów mobilnych (2D) pozwalającą na planowanie ruchu robota w czasie rzeczywistym [43]. Dlatego postanowiono wykorzystać jej możliwości i zaadaptować ją do rozwiązania zadania wyznaczenia kierunku obejścia przeszkód w przestrzeni 3D.

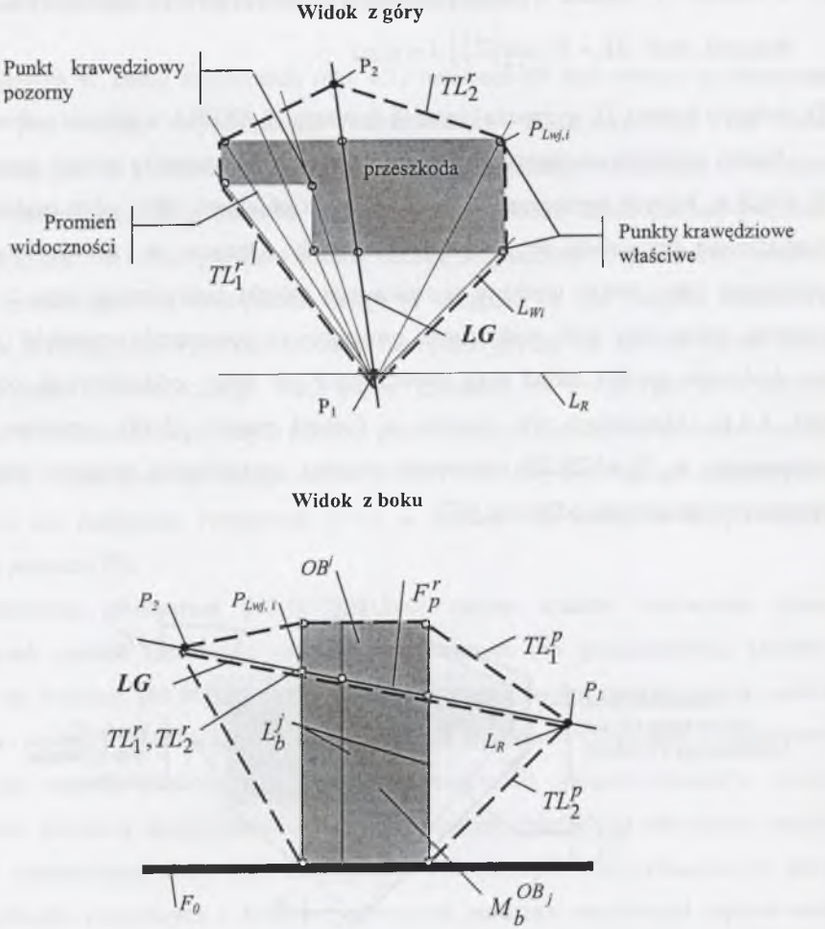
Algorytm strategii pomocniczej ψ_2 przebiega następująco (rys. 5.15) [97,98]:

1) przez P_1 poprowadzić prostą L_R prostopadłą do LG i równoległą do płaszczyzny podstawy sceny robota F_o. Proste LG i L_R wyznaczają płaszczyznę planowania F_p^r ścieżki robota. Płaszczyzna F_p^r przecina obiekt OB^j w kierunku LG. Przecięcie płaszczyzny F_p^r z krawędziami OB^j wyznacza tzw. punkty krawędziowe $P_{Lw_{j,i}}$ obiektu OB^j: $P_{Lw_{j,i}} = \sum_j \sum_i (F_p^r \cap b_l^i)$; gdzie: b_l^i, l - pkt 5.2.4.1;

2) wyznaczyć proste $L_{w_i} = L(P_1, P_{Lw_{j,i}})$ leżące w płaszczyźnie F_p^r, przechodzące przez P_1 i przez każdy z wyznaczonych punktów $P_{Lw_{j,i}}$. Proste te tworzą pęk, którego P_1 jest wierzchołkiem. Proste L_{wi} przecinają obiekt j, do którego należą wyznaczone punkty $P_{Lw_{j,i}}$. Jeżeli $L_{w_i} \cap OB^j > 1$ oznacza to, że punkt krawędziowy wskazujący prostą L_{wi} leży „w świetle obiektu OB^j" – jest tzw. punktem krawędziowym pozornym. Jeżeli zaś $L_{w_i} \cap OB^j = 1$, to jest to punkt krawędziowy właściwy;

3) postępując jak w pkt. 1 i 2 wyznaczyć punkty krawędziowe właściwe leżące na płaszczyźnie F_p^p prostopadłej do płaszczyzny F_0 przechodzącej przez LG;

4) uporządkować znalezione punkty krawędziowe właściwe, porównując ich współrzędne, tak by pozostawały one w relacji widoczności zachodzącej pomiędzy sobą, na leżące po jednej i drugiej stronie prostej LG i równocześnie leżące na płaszczyznach F_p^r i F_p^p



Rys. 5.15. Strategia ψ_2 [186]

Fig. 5.15. Strategy ψ_2 [186]

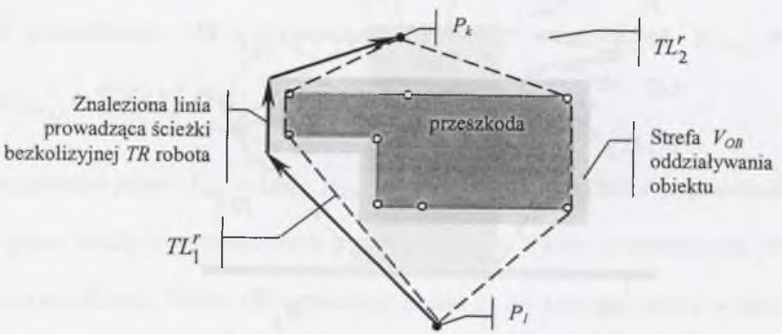
(pomijane są te, które nie przynależą do przestrzeni roboczej robota V_{RP} (5.20), czyli nie spełniają warunku: $\forall_j \forall_i P_{Lw_{j,i}} \in \Pi_{RP}$);

5) poczynając od położenia P_1, poprowadzić odcinki łącząc kolejno wyznaczone punkty graniczne właściwe (promienie widoczności) leżące na F_p^r i F_p^p po jednej i po drugiej stronie LG, aż do P_k. Odcinki te tworzą zbiór TL tzw. łamanych prowadzących: TL_1^r, TL_2^r i TL_1^p, TL_2^p;

6) w zbiorze TL znaleźć tę łamaną prowadzącą, która spełnia przyjęty warunek minimalnej długości, czyli: $TL = \forall_{l} \min_{i=1,2}\left(TL_i^l\right),\ l = r, p$;

7) wybrana łamana TL wyznacza kierunek działania PLANERA w oparciu o strategię ψ_1.

Należy wyraźnie zaznaczyć, że znaleziona łamana TL wyznacza jedynie granice działania PLANERA, których przekroczenie grozi kolizją z obiektem OB^j, gdyż znalezione punkty krawędziowe przynależą do omijanej przeszkody. Oznacza to, że nie mogą one być traktowane jako punkty węzłowe poszukiwanej ścieżki bezkolizyjnej (rys. 5.16). Z tego powodu, w metodzie grafu widoczności proponuje się poszerzenie przeszkód [43]. Wymóg ten doskonale spełnia układ wag wynikających ze strefy oddziaływania obiektu sceny (pkt 5.4.1). Mechanizm ten, łącznie z funkcją nagród (5.38), powinien umożliwić wymuszenie na PLANERZE odsunięcie procesu poszukiwania punktów węzłowych na dostatecznie bezpieczną odległość [97].



Rys. 5.16. Linia prowadząca planowanej ścieżki robota

Fig. 5.16. Line leading to the planned path of the robot

Wpływ strefy oddziaływania obiektu na ruch robota sprawi więc, że typowanie kolejnych rastrów wyznaczających stany osiągane przez robot przy omijaniu przeszkody będą leżały co najwyżej na skraju obszaru oddziaływania V_{OB}. Strefa ta spowoduje „odsunięcie" punktów krawędziowych właściwych na skraj strefy V_{OB}, tym samym wyznaczając zmodyfikowany układ promieni widoczności wyznaczających linię prowadzącą PLANER, wg której typowane będą przez niego kolejne punkty węzłowe.

5.7. OGRANICZENIA UZYSKIWANYCH ROZWIĄZAŃ

W przyjętych w pracy założeniach (pkt 4.1) ustalono, że ruch robota w przestrzeni opisywany jest trójką współrzędnych kartezjańskich $P_{wi}(x_i,y_i,x_i) \in U_{RP}$, zaś robot identyfikowany jest z punktem charakterystycznym manipulacji związanym z ostatnim elementem ostatniego ogniwa łańcucha kinematycznego robota. Równocześnie zwrócono uwagę (5.20), pkt 5.2.1, że typowane punkty muszą przynależeć do przestrzeni roboczej robota, by poruszający się manipulator mógł je osiągnąć, a tym samym zrealizować zaplanowaną ścieżkę. Dla typowych robotów manipulacyjnych, dla których liczba stopni swobody łańcucha kinematycznego waha się w granicach od $r=4\text{-}6$ (pkt 2.2), przyjęte ograniczenie w definiowaniu kolejnych położeń przestrzennych sprawia, że zachodzi warunek redundantności układu kinematycznego robota [43,50,73,80,140], czyli łańcuch kinematyczny manipulatora ma nadmierną ruchliwość ($r>3$) w stosunku do wymiarowości przestrzeni, w której się porusza (R^3).

W literaturze robotycznej [43,50,73,80,140] można znaleźć omówienie metod rozwiązujących zadanie kinematyki robotów redundantnych (np. pseudoinwersji Moore'a-Penrose'a lub ważonej [43,80]), w których określone są sposoby rozwiązywania zadania odwrotnego i warunki, jakie muszą zostać spełnione, by było ono wyznaczalne. Przykładowo, wykorzystując metodę psudoinwersji (pseudoodwracalności) Moore'a-Penrose'a, można zweryfikować, uzyskany drogą odpowiednich przekształceń, opartych na odwrotnym modelu kinematyki manipulatora, zbiór jego konfiguracji wewnętrznych dla wyznaczonych przez PLANER położeń związanych z kolejno typowanymi punktami węzłowymi zaplanowanej ścieżki. By wektor współrzędnych uogólnionych q_{i+1} związany z osiąganym stanem $s_{i+1}(P_{wi})$ był możliwy do zrealizowania, konieczne jest spełnienie warunku, że [43,80]:

$$\dot{q}_{i+1} = J^{\#}(q_{i+1}) \cdot \dot{d}_{i+1} \tag{5.46}$$

gdzie: $J^{\#}(q_{i+1}) = J^{T_Q}(q_{i+1}) \cdot \left[J(q_{i+1}) \cdot J^{T_Q}(q_{i+1}) \right]^{-1}$ – oznacza pseudoodwracalność Moore'a-Penrose'a,

$J(q_{i+1})$ – jakobian określony dla położenia (konfiguracji) d_{i+1} wyrażony wektorem współrzędnych konfiguracyjnych w wewnętrznej przestrzeni stanu robota dla zadanego położenia P_{wi}, zdefiniowanego w przestrzeni zadania,

$J^{T_Q}(q_{i+1})$ – transponowany jakobian $J(q_{i+1})$ w przestrzeni stanów wewnętrznych Q,

d_{i+1} – wektor położenia określony dla P_{wi+1},

q – wektor współrzędnych uogólnionych określony w wewnętrznej przestrzeni stanu Q dla zadanego d_{i+1},

$\dot{d}_{i+1}$ – pochodna przemieszczenia robota do P_{wi+1} względem czasu,

$\dot{q}_{i+1}$ – pochodna wektora współrzędnych uogólnionych q określonego dla P_{wi+1} względem czasu.

Równocześnie musi być spełniony warunek, że macierz przekształcenia prostego T kinematyki manipulatora transponuje konfigurację robota z przestrzeni zadania do wewnętrznej przestrzeni stanu [43,80]:

$$T_Q^{-1}: d_{i+1}(s_{i+1}) \quad q_{i+1}=T_Q^{-1}(d_{i+1}) \tag{5.47}$$

oraz:

$$d_{i+1}(s_{i+1}) \in \Pi_{RB} \qquad \text{i} \qquad q_{i+1} \in Q.$$

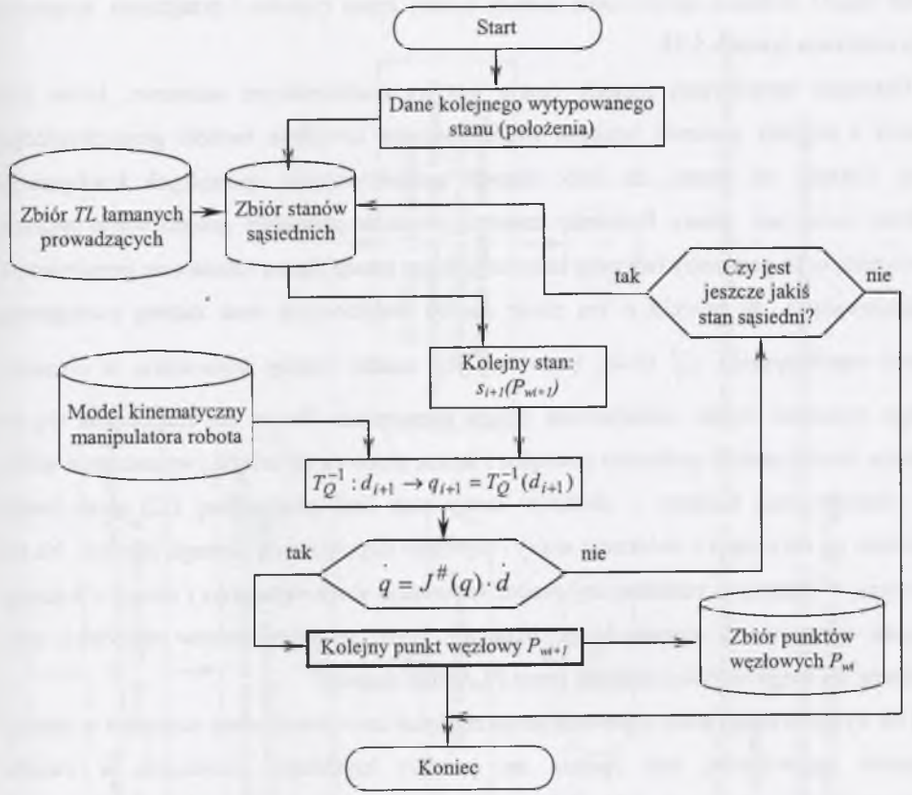
gdzie: T_Q^{-1} – macierz przekształcenia odwrotnego zadania kinematyki,

Π_{RB} – przestrzeń konfiguracji zewnętrznej robota (kartezjańska przestrzeń zadania),

Q – wewnętrzna przestrzeń konfiguracyjna,

czym spełnia warunek przekształcenia odwrotnego zadania kinematyki.

Na rysunku 5.17 pokazano przebieg procedury weryfikacji typowanych przez PLANER stanów. Jeżeli planowany kolejny stan $s_{i+1}(P_{wi+1})$ spełnia warunki (5.46) i (5.47), to związane z nim położenie w przestrzeni może uzupełnić istniejący zbiór punktów węzłowych planowanej ścieżki. Jeżeli zaś ich nie spełnia, to w oparciu o zachowany, mimo ograniczeń narzuconych strategiami lokalnymi ψ_1 i ψ_2 i korytarzem planowania (parametr C_T), efekt błądzenia (pkt 5.6.1.1) konieczne jest wytypowanie dla aktualnego miejsca bieżącego innego stanu spośród stanów sąsiednich PLANERA lub wykorzystanie innej wyznaczonej ścieżki ze zbioru TL (strategia ψ_2, pkt 5.6.1.2).



Rys. 5.17. Weryfikacja typowanych położeń przestrzennych planera

Fig. 5.17. Verification of the specified spatial positions of the planner

Zastosowana do weryfikacji typowanych położeń planowanej ścieżki bezkolizyjnej metoda psudoinwersji Moore'a-Penrose'a pozwala również zweryfikować tzw. stany osobliwe konfiguracji manipulatora w wyznaczonych miejscach w przestrzeni [43,80].

5.8. ALGORYTM LOGICZNY OPRACOWANEJ METODY

Opracowana metoda, w założeniu, przeznaczona jest do wspomagania procesu programowania robotów przemysłowych metodą off-line. Wieloetapowy tok postępowania, wynikający ze specyfiki rozwiązywanych problemów sprawia, że celowe wydaje się uporządkowanie wszystkich elementów metody w ciąg logiczny tworzący jej algorytm, dzięki

czemu zasada działania opracowanej metody byłaby lepiej czytelna i przejrzysta. Algorytm ten przedstawia rysunek 5.18.

Działanie opracowanej metody oparte jest na podstawowym założeniu, jakim jest ustalony i przyjęty warunek istnienia zdefiniowanego formalnie modelu geometrycznego sceny. Zakłada się zatem, że zbiór danych geometrycznych opisujących konfigurację obiektów sceny jest znany. Podobnie znane są pozostałe parametry geometryczne systemu planowania, czyli parametry łańcucha kinematycznego manipulatora robota oraz przedmiotów manipulowanych. W oparciu o ten zbiór danych wejściowych oraz zadaną początkową wartość współczynnika C_1^j (5.40) budowany jest model oktalny środowiska, w obszarze którego zachodzić będzie zainicjowany proces planowania. Proces ten rozpoczyna się od ustalenia współrzędnych położenia początku i końca planowanej ścieżki, wyznaczenia strefy V_{OB} oddziaływania każdego z obiektów sceny oraz linii geodezyjnej *LG*, sprawdzenia warunków jej incydencji z obiektami sceny i wybranie odpowiedniej strategii lokalnej. Na tej podstawie, w oparciu o zdefiniowany proces sterowania stochastycznego i strategie lokalne, następuje wytypowanie odpowiedniego oktantu - stanu, ze zbioru stanów sąsiednich oraz określenie dla niego wartości zdobytej przez PLANER nagrody.

Dla wytypowanego stanu zapewniającego zdobycie najwyższej oceny następuje w dalszej kolejności sprawdzenie, czy spełnia on przyjęte kryterium jakościowe w świetle obowiązujących ograniczeń (pkt 5.7) wynikających z redundantności uzyskiwanych rozwiązań. Jeżeli wszystkie określone metodą warunki zostaną spełnione, typowany stan przyjęty zostaje jako kolejny punkt węzłowy planowanej ścieżki. Proces planowania rozpoczyna kolejną iterację aż do osiągnięcia stanu końcowego.

Zaprezentowany algorytm posłużył do opracowania programu Wiz_SRM symulującego pracę PLANERA, który pozwolił na ocenę postawionych w pracy założeń, weryfikację opracowanej metody i sprawdzenie zasadności określonej w pracy tezy.

5.8.1. Wiz_SRM - symulator PLANERA bezkolizyjnych ścieżek robota

Program *Wiz_SRM* opracowano wykorzystując jeden z istniejących na rynku systemów programowania, wspomagających wizualizację zagadnień geometrycznych. Program przeznaczony jest do implementowania na komputerach klasy *PC* wykorzystujących środowisko Windows. Podstawowym kryterium tworzonego oprogramowania symulatora opracowanego PLANERA było założenie jego maksymalnego uproszczenia. Przyjęto mianowicie, że podstawowym celem programu jest jedynie sprawdzenie poprawności

Rys. 5.18. Schemat ideowy działania PLANERA bezkolizyjnych ścieżek robota stacjonarnego i manipulacyjnego w zewnętrznej przestrzeni zadania

Fig. 5.18. Schematic diagram illustrating the operation of the PLANNER of collision-free stationary and manipulative robot routes in the external space of the task

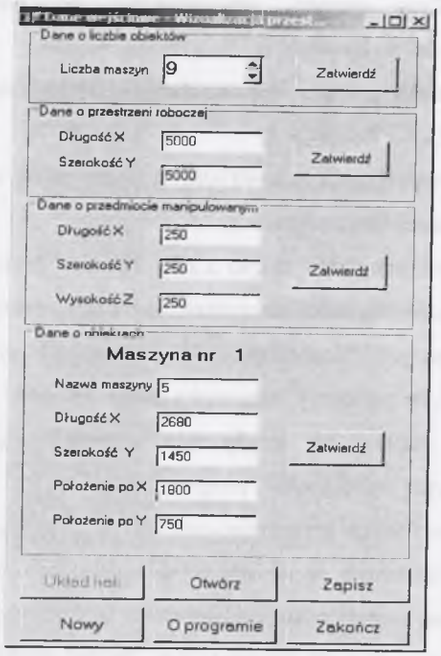
opracowanej metody. Tym samym zrezygnowano ze wszystkich udogodnień programistycznych, których celem byłoby:

- udogodnienie wprowadzania i wyprowadzania danych, większe niż jest to niezbędne do zweryfikowania metody,
- umożliwienie importowania danych geometrycznych sceny robota z programu typu CAD,
- poprawa ekspresji wizualnej ekranu (grafiki).

Ponieważ zgodnie ze wzorem Stirlinga (pkt 3.2.2.1.1) o szybkości działania opracowanej metody nie decyduje szybkość procesora komputera, na którym zostanie ona zweryfikowana, lecz wyłącznie liczba operacji niezbędnych do rozwiązania postawionego problemu planowania, zatem przyjęto, że parametry stanowiska komputerowego nie wpływają na jakość uzyskiwanych wyników; ważne, by mogły one sprostać wymaganiom środowiska programowego. Tym samym opracowany program mógł spełnić postawione założenie prostoty zastosowanych rozwiązań programistycznych. W rezultacie otrzymano proste i łatwe w obsłudze oraz skuteczne w działaniu narzędzie pozwalające na przebadanie wszystkich elementów opracowanej metody planowania bezkolizyjnych ścieżek robotów o minimalnych wymaganiach sprzętowych (procesor min. 700 kH, RAM 512 kB).

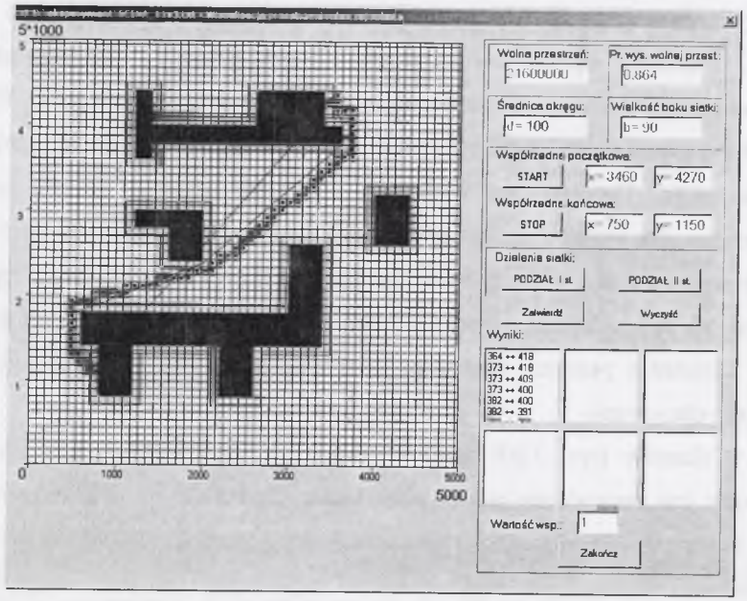
Przedstawiony algorytm logiczny metody (rys. 5.18) posłużył pierwotnie do opracowania szczegółowej struktury omawianego programu Wiz_SRM i algorytmów modułów programowych realizujących odpowiednie zadania logiczne modelowanego procesu planowania. Działanie opracowanego programu sprowadza się do interaktywnego współdziałania z operatorem w obszarze dwóch okien programowych (ekranów). Pierwsze z nich (rys. 5.19) przeznaczone jest do zadawania danych, wg których symulowany ma być proces planowania. Dotyczą one wielkości przestrzeni roboczej robota, przedmiotu manipulowanego oraz współczynnika korekcyjnego strefy oddziaływania obiektu i stanowią model geometryczny sceny, w której badany jest proces planowania. Z poziomu tego ekranu możliwe jest ich eksportowanie w postaci pliku (*.geo) do bazy danych geometrycznych programu. Również z poziomu tego okna dane te mogą być z powrotem importowane do kolejnych sesji planowania.

Drugi z ekranów (rys. 5.20) zawiera model 2D zdefiniowanej sceny, na którym wizualizowany jest prowadzony proces planowania. Umożliwia on definiowanie położeń początku i końca planowanej ścieżki robota oraz ilustruje uzyskane wyniki, łącznie z oknem danych (*Memo*) zawierającym wygenerowane współrzędne punktów węzłowych P_{wi} znalezionej ścieżki bezkolizyjnej. Pokazuje również wszystkie ważne parametry realizowanego procesu sterowania stochastycznego, takie jak: objętość wolnej przestrzeni



Rys. 5.19. Ekran wprowadzania danych programu Wiz_SRM

Fig. 5.19. Screen of input data for Wiz_SRM program



Rys. 5.20. Ekran pola symulacji procesu planowania

Fig. 5.20. Screen of simulation field for the planning process

w której zachodzi proces planowania, wartość współczynnika korekcyjnego C_k (5.10), które są niezbędne do określenia parametrów oktalnego podziału startowego b^s (5.9). Ekran ten umożliwia również podgląd miejsc, w których może wystąpić konieczność dokonania podziałów oktalnych na wyższych niż poziom startowy poziomach. Służą do tego celu opcje *Podział I st.* i *Podział II st.* Założono, że ze względu na szczegółowość przedstawienia tej możliwości na ekranie monitora, będzie to możliwe tylko dla podziałów dokonywanych na dwóch kolejnych poziomach, począwszy od poziomu startowego.

6. Weryfikacja metody. Eksperyment symulacyjny

Celem sprawdzenia stopnia spełnienia postawionej w pracy tezy (pkt 4) przeprowadzono - drogą eksperymentu symulacyjnego - weryfikację uzyskanych rozwiązań. Otrzymane i przedstawione rezultaty przeprowadzonych symulacji komputerowych umożliwiły porównanie otrzymanych wyników z założeniami pracy oraz oczekiwaniami stawianymi opracowanej metodzie, szczególnie w kontekście wykorzystania jej na etapie programowania off-line robotów stacjonarnych i manipulacyjnych.

Badania symulacyjne miały za zadanie praktyczne udowodnienie, że opracowana strategia lokalna $\Psi : \psi_1 \cup \psi_2$ w sposób istotny, pozwala na ograniczenie przestrzeni poszukiwań V i efektywne (tj. szybkie) poszukiwanie rozwiązań dopuszczalnych.

Dodatkowo oczekiwano się, że opracowana metoda pozwoli wyeliminować z procesu planowania takie niepożądane zjawiska, jak minima lokalne i pętle, typowe dla procesu błądzenia.

W opracowanym programie pominięto zagadnienie więzów nakładanych na planowaną trajektorię, gdyż przyjęto, że problem ten wymaga osobnego potraktowania, a zakres opracowanej metody ogranicza się do umiejętności radzenia sobie PLANERA w złożonym geometrycznie środowisku sceny robota. Dlatego przyjęto a priori, że wszystkie wyznaczone przez PLANER punkty węzłowe planowanych ścieżek są możliwe do zrealizowania, czyli że spełniają warunek (5.20).

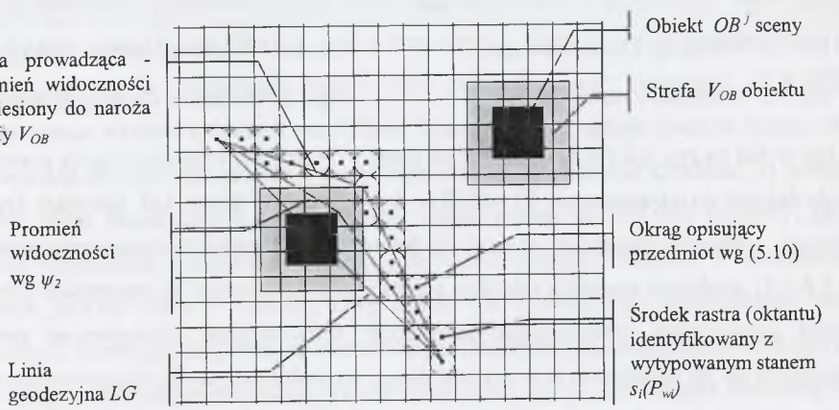
6.1. EKSPERYMENT SYMULACYJNY

6.1.1. Dane

Przy doborze danych do przeprowadzenia symulacji przyjęto, że powinny one w możliwie największym stopniu odpowiadać typowym warunkom, w jakich pracują przemysłowe roboty manipulacyjne i stacjonarne. Dlatego przyjęty w symulacji obszar działań PLANERA obejmuje przestrzeń sceny, w której znajdują się rozmieszczone przypadkowo obiekty OB^j.

Dane geometryczne o scenie robota zostały wprowadzone do programu w postaci pliku zawierającego pełną informację o scenie robota i jej uwarunkowaniach geometrycznych (wielkość przestrzeni roboczej, liczba obiektów sceny, ich wymiary i wzajemne rozmieszczenie oraz wielkość przedmiotu manipulowanego). Pozwalają one na wyliczenie wymaganych w procesie planowania parametrów, takich jak: wielkość boku siatki oktalnej, wielkość przestrzeni bezkolizyjnej oraz współczynników korekcyjnych, wag i wartości funkcji prawdopodobieństw warunkowych wymaganych w zastosowanym procesie sterowania stochastycznego.

Wyniki przeprowadzonych symulacji prezentowane są w postaci widoku ekranu, a zawarty na nim układ informacji, które ułatwiają interpretację danych i uzyskanych wyników, pokazano na rys. 6.1.



Rys. 6.1. Elementy ekranu symulatora

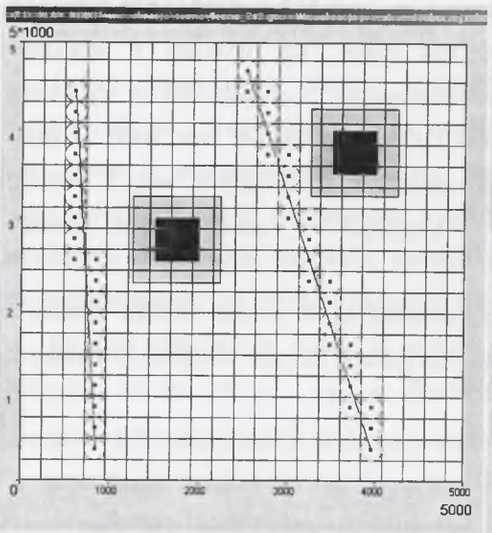
Fig. 6.1. Elements of the screen of the simulator

Wszystkie przeprowadzone próby symulacyjne mają za zadanie zilustrowanie zachowania się PLANERA w warunkach określonych konfiguracją przestrzenną sceny,

przedmiotem manipulowanym oraz parametrami położenia początkowego i końcowego poszukiwanej bezkolizyjnej ścieżki *TR* robota.

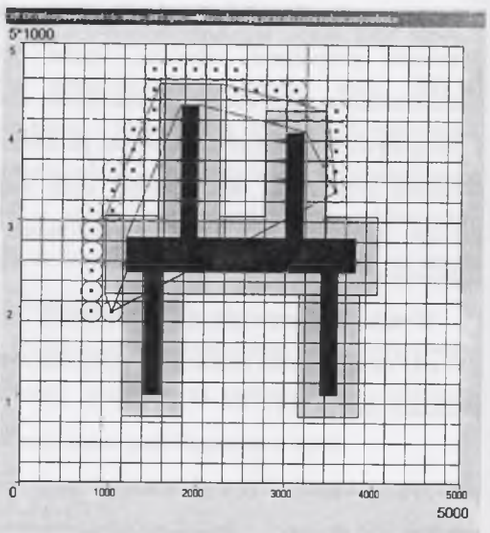
6.1.2. Wyniki symulacji

Pierwszym elementem przeprowadzonego testu symulacyjnego było sprawdzenie poprawności działania opracowanych (pkt 5.6.1) strategii lokalnych ψ_1 (rys. 6.2) i ψ_2 (rys. 6.3).



Rys. 6.2. Weryfikacja działania strategii ψ_1

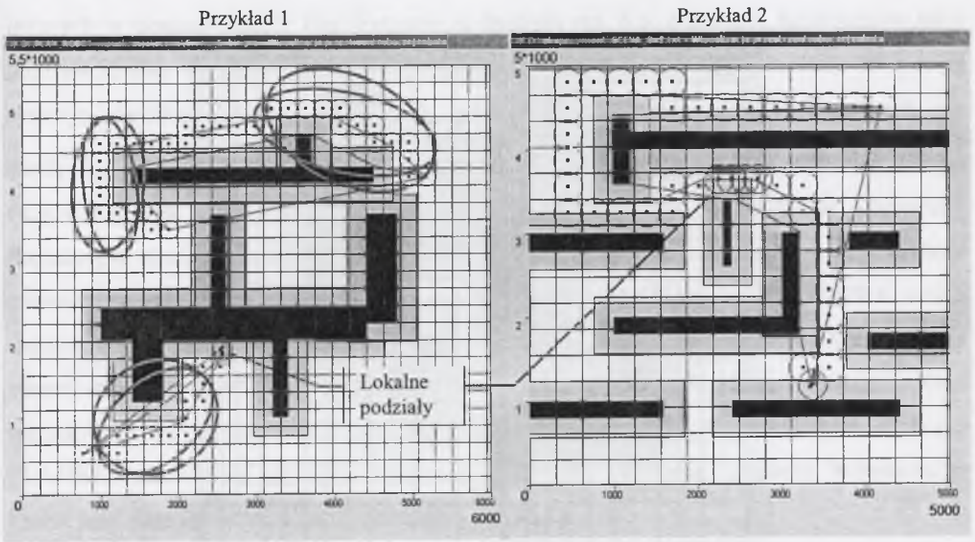
Fig. 6.2. Verification if strategy ψ_1 works



Rys. 6.3. Weryfikacja działania strategii ψ_2

Fig. 6.3. Verification if strategy ψ_2 works

Jak widać na rys. 6.2 i 6.3, wyznaczone przez PLANER ścieżki przebiegają prawidłowo. Istnieje dążenie do trzymania się PLANERA wyznaczonego przez *LG* kierunku (rys. 6.2). W sytuacji ominięcia napotkanej przeszkody promienie widoczności wyznaczone strategią ψ_2 (pkt 5.6.1.2) właściwie spełniają rolę linii prowadzącej. Omówiony w przypadku strategii ψ_2 problem konieczności powiększania przeszkody, spowodowany zaczepieniem promienia widoczności do jej naroża, rozwiązany jest drogą wyznaczonej strefy oddziaływania obiektu V_{OB} i związanej z nią wagi C_2. Jak widać na rys. 6.2 i 6.3 oraz wszystkich kolejnych, z każdym z wyznaczonych promieni widoczności związany są ich odpowiedniki „zaczepione" w narożach strefy oddziaływania obiektu V_{OB} wyznaczające linię prowadzącą (pkt 5.6.1.2), względem której następuje planowanie poszukiwanej ścieżki bezkolizyjnej w oparciu o ψ_1. Tym samym uzyskano skuteczne narzędzie, pozwalające na odsunięcie procesu typowania

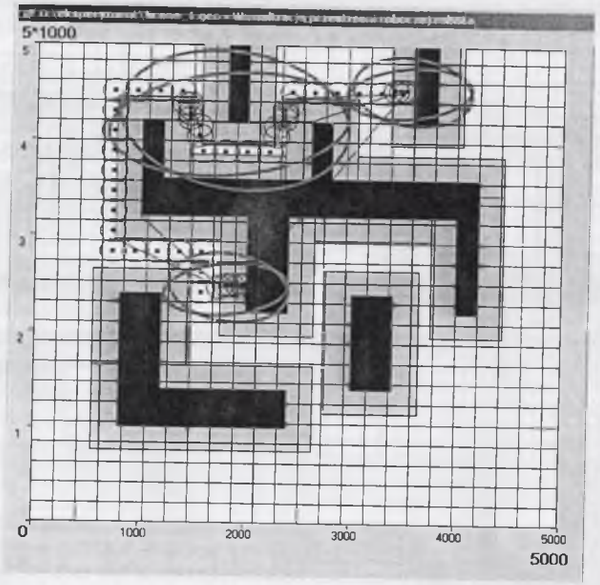


Rys. 6.4. Wpływ strefy V_{OB} oddziaływania obiektów na swobodę poruszania się robota w przestrzeni

Fig. 6.4. Impact of sphere V_{OB} on the interactions between objects and free motion of the robot in space

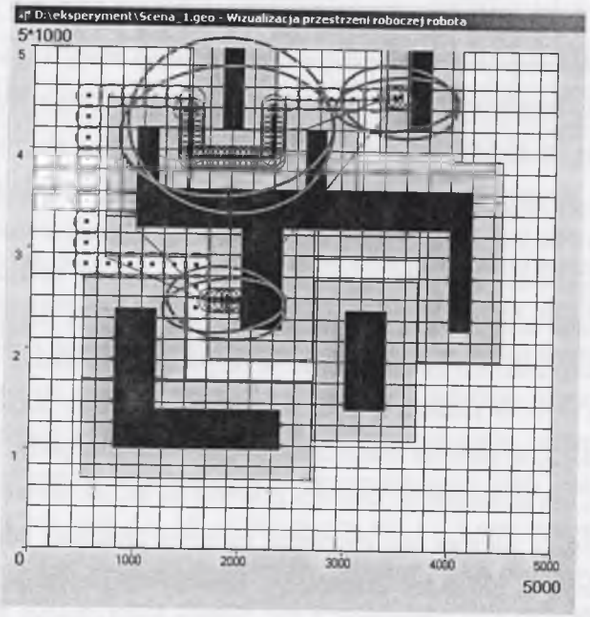
przez PLANER kolejnego punktu węzłowego na odległość bezpieczną od każdej napotkanej przeszkody. O bliskości przejścia względem obiektu decyduje środek geometryczny rastra, co ilustruje rys. 6.4.

Dodatkowo, tak interpretowane oddziaływanie omawianej strefy V_{OB} i związanej z nią wagi C_1 powoduje, że obszar wokół obiektu, zajęty przez nią, staje się niedostępny do wykorzystania przez PLANER (rys. 6.4) i tym samym niemożliwy do penetrowania przez poruszającego się w przestrzeni robota obok obiektu sceny, zmuszając PLANER do wygenerowania lokalnie kolejnych podziałów. Sytuacje takie ilustrują również rysunki 6.5 do 6.7. Ponieważ przyjęto, że strefa oddziaływania obiektu ma jedynie utrudniać jej penetrację (chroniąc robot przed kolizją z przeszkodą), zatem dostęp do niej jest możliwy, ale pod pewnymi warunkami. Ustalono mianowicie, że w bezpośredniej bliskości obiektu może się znajdować jedynie robot w sytuacji, kiedy obiekt ten wymaga jego bezpośredniej obsługi. Tym samym określono, że jedynym warunkiem geometrycznym, na podstawie którego może taka sytuacja zajść, jest prostopadłość kierunku zbliżenia się do, jak również oddalania się od obiektu. Wydaje się, że warunek taki gwarantuje możliwie najkrótszą drogę poruszania się robota w bezpośredniej bliskości przeszkody. Dodatkowo przyjęto, że sytuacja taka musi zachodzić z zachowaniem największej precyzji działania robota. Dlatego też każdy ciąg ruchów robota odbywa się przy spełnieniu dodatkowych podziałów oktalnych rastrów



Rys. 6.5. Wymuszenie oktalnych podziałów lokalnych strefą V_{OB} sąsiadujących ze sobą obiektów

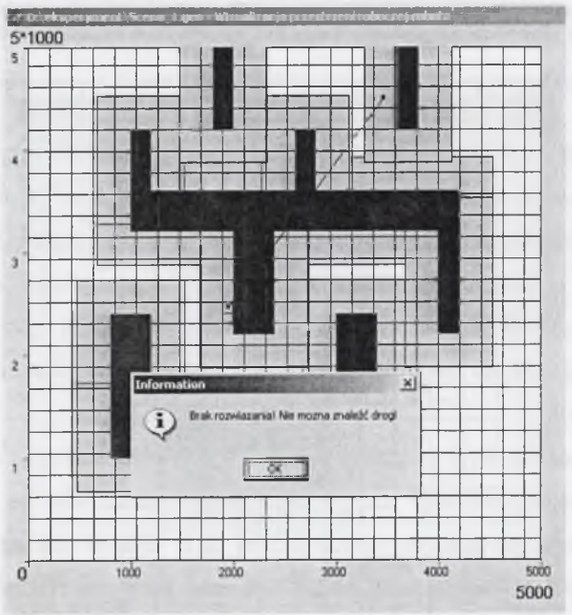
Fig. 6.5. Input of octal local divisions of contiguity objects by V_{OB} sphere



Rys. 6.6. Spowolnienie działania PLANERA spowodowane bliskością sąsiadujących ze sobą obiektów sceny

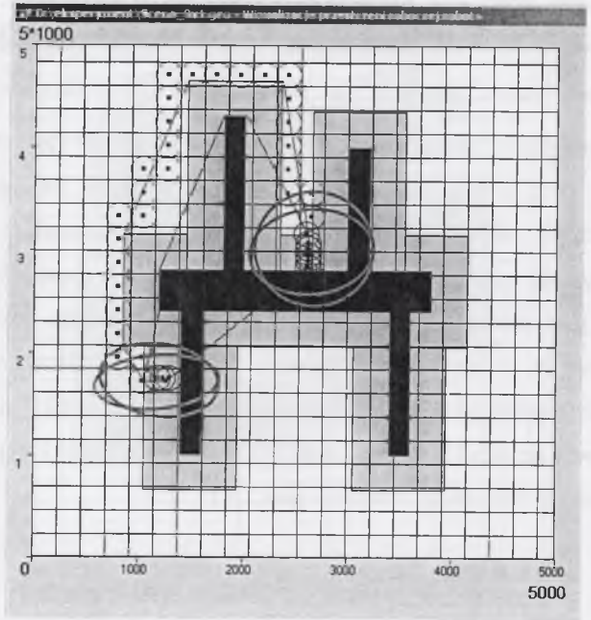
Fig. 6.6. Deceleration in the PLANNER operation caused by the vicinity of contiguity object of the operation scene

leżących w obszarze strefy V_{OB}. Sytuację tę ilustrują rys. 6.5, 6.6 i 6.8. Rozwiązanie takie pozwala spełnić warunki, wynikające z zagęszczenia przestrzeni obiektami sceny OB^j, czyli sytuację określoną przez C_k (5.10), C_{OB^j} (5.40) i C_2 (5.42). W skrajnych przypadkach, gdy strefa V_{OB} jest określona nieprawidłowo, może to doprowadzić do zablokowania działań PLANERA (rys. 6.7), a tym samym uniemożliwić poruszanie się robota w przestrzeni. Pożądanym skutkiem takiego rozwiązania jest to, że zagęszczenie sceny przeszkodami prowadzi w oczywisty sposób do znaczącego wzrostu precyzji działania PLANERA, a tym samym bardziej precyzyjnego – ostrożnego – poruszania się robota w przestrzeni (gradient planowania decyduje o precyzji planowania, pkt 5.2.4.2). Skutkiem ubocznym tego mechanizmu jest spowolnienie procesu planowania. Jednakże taki mechanizm działania PLANERA pojawia się jedynie w obszarach, w których typowanie kolejnego bezpiecznego kroku jest zagrożone bliskością obiektów (pkt 5.5.1).



Rys. 6.7. Zmiana układu geometrycznego przestrzeni bezkolizyjnej spowodowana strefą V_{OB}

Fig. 6.7. Change in the geometric system of collision-free space evoked by sphere V_{OB}

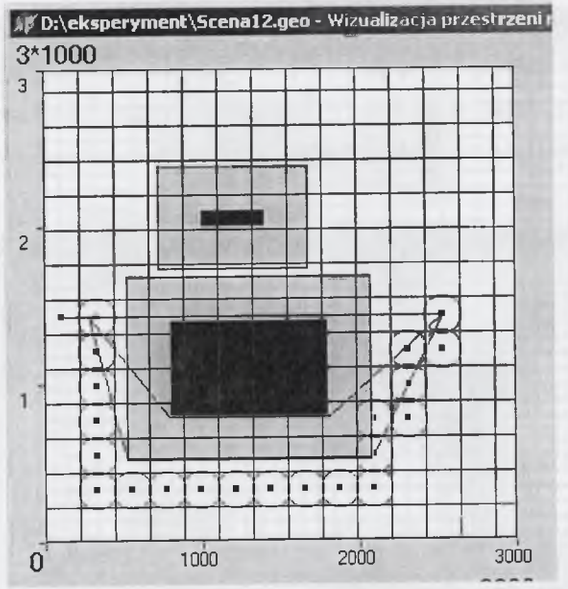


Rys. 6.8. Zagęszczenie gradientu planowania przy zbliżaniu się robota do obiektu

Fig. 6.8. Compression of the planning gradient while the robot approximates an object

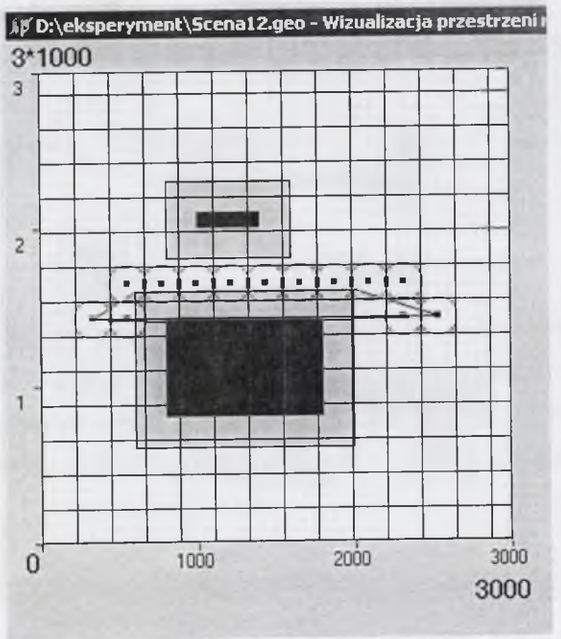
Jak widać na zaprezentowanych rys. 6.5 i 6.6, również bezpośrednia bliskość sąsiadujących ze sobą obiektów wywołuje efekt „zagęszczenia" typowanych punktów węzłowych P_{wi}, będącego wynikiem zmniejszenia gradientu planowania (5.11) i tym samym skrócenia kroków t_i pomiędzy kolejnymi punktami węzłowymi. Jednak działanie wagi C_2, zniechęcającej PLANER do wchodzenia w obszary wąskie, a tym samym unikania zwiększonej szczegółowości planowania, jest dla PLANERA zasadniczym działaniem. Efekt ten pokazuje rys. 6.9.

W sytuacji kiedy w myśl przyjętego kryterium optymalnościowego najkrótszej drogi i ogólnej strategii maksymalizacji zdobywanych nagród okaże się, że przejście wąskim przesmykiem istniejącym pomiędzy przeszkodami da w efekcie znaczące skrócenie drogi, nawet kosztem zaniżonej oceny, to PLANER wybierze takie rozwiązanie (rys. 6.9, 6.10, 6.11 i 6.12). Rozwiązanie to jest tym pewniejsze, im większa jest możliwość przejścia bez dokonania dodatkowych podziałów w przesmyku pomiędzy obiektami. Na rys. 6.12 przejście pomiędzy obiektami jest możliwe również dzięki temu, że gradient planowania jest odpowiednio mały, a odległość pomiędzy obiektami spełnia warunek (5.41).



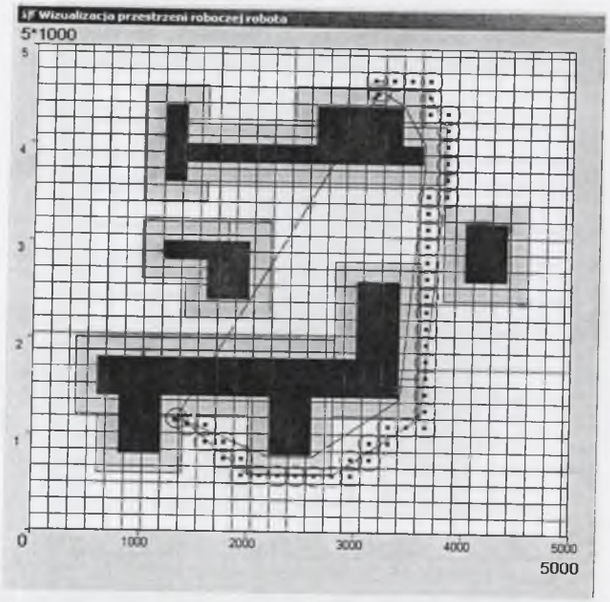
Rys. 6.9. Wybór dłuższej ścieżki spowodowany strategią maksymalizacji nagród

Fig. 6.9. Selection of a longer path due to the reward maximization strategy



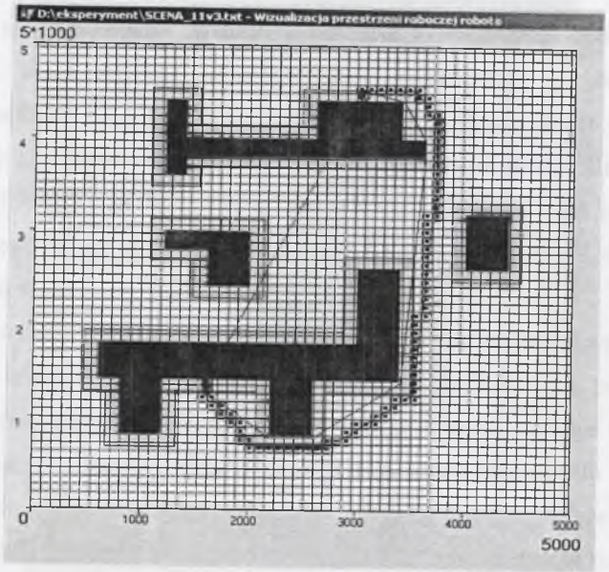
Rys. 6.10. Wpływ kryterium najkrótszej drogi na wybór ścieżki optymalnej

Fig. 6.10. Influence of the shortest path criterion on the selection of the optimal route



Rys, 6.11. Planowanie ścieżki bezkolizyjnej w przesmyku pomiędzy obiektami

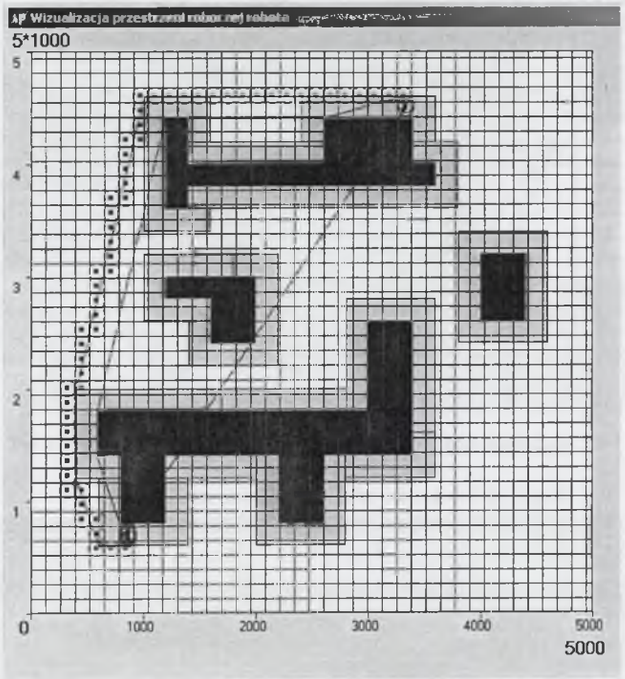
Fig. 6.11. Planning of collision free path in the passage between the objects



Rys. 6.12. Wpływ gradientu planowania na przebieg planowanej ścieżki

Fig. 6.12. Impact of the planning gradient of the course of the planned path

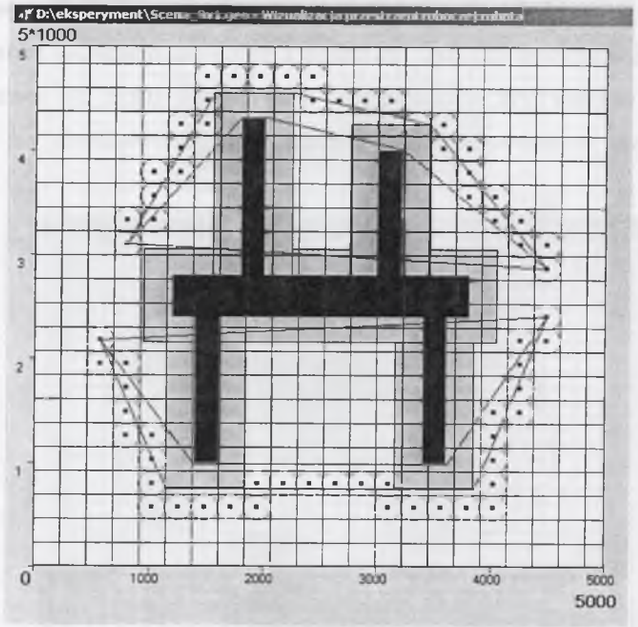
Trzymanie się przez PLANER linii prowadzącej pozwala jednoznacznie wyeliminować wszystkie jego nieefektywne działania związane z procesem błądzenia, takie jak minima lokalne i pętle, nawroty (rys. 6.13 i 6.14). Wynika to z faktu, że wyznaczenie linii prowadzącej i korytarza planowania ogranicza do minimum efekt błądzenia.



Rys. 6.13. Eliminacja minimów lokalnych

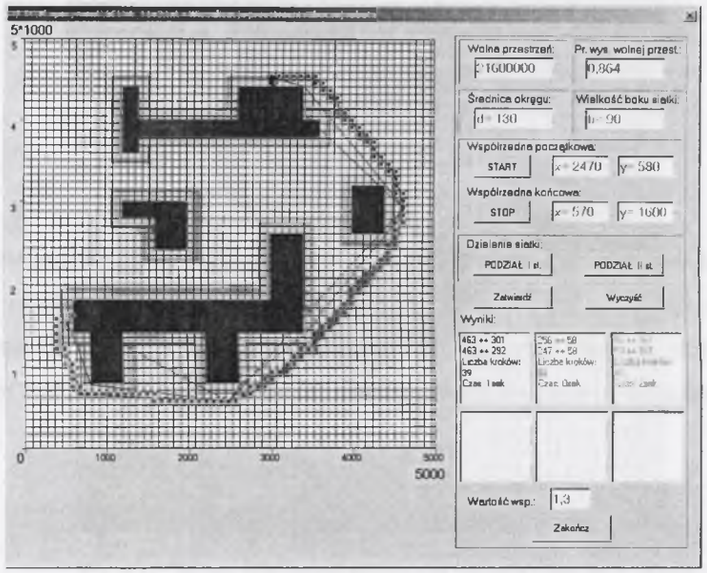
Fig. 6.13. Elimination of local minimums

Dzięki odpowiedniemu programistycznemu rozwiązaniu możliwe jest generowanie przez PLANER złożonych ścieżek powstających jako efekt procesu „sklejania" kilku prostszych. Mechanizm taki pozwala uzyskać efekt „wymuszenia" na PLANERZE poszukiwania rozwiązania wskazanego przez programistę. Przypadki takie ilustrują rys. 6.15, 6.16 i 6.17.



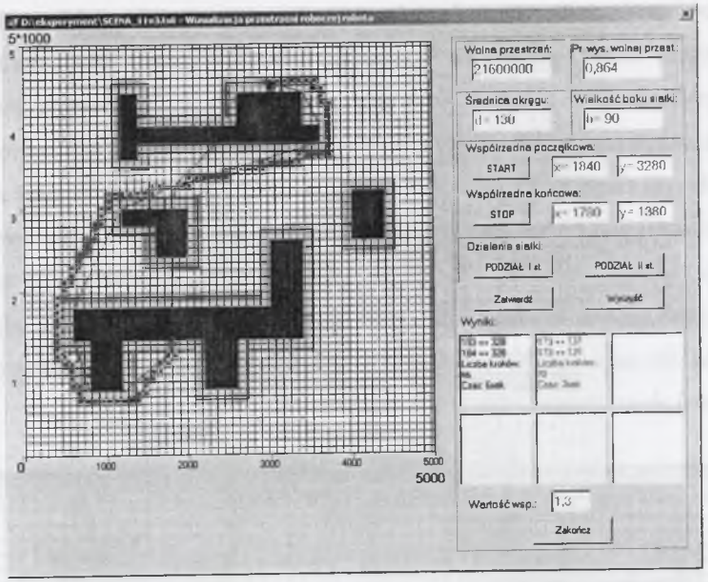
Rys. 6.14. Eliminowanie minimów lokalnych i pętli

Fig. 6.14. Elimination of local minimums and loops



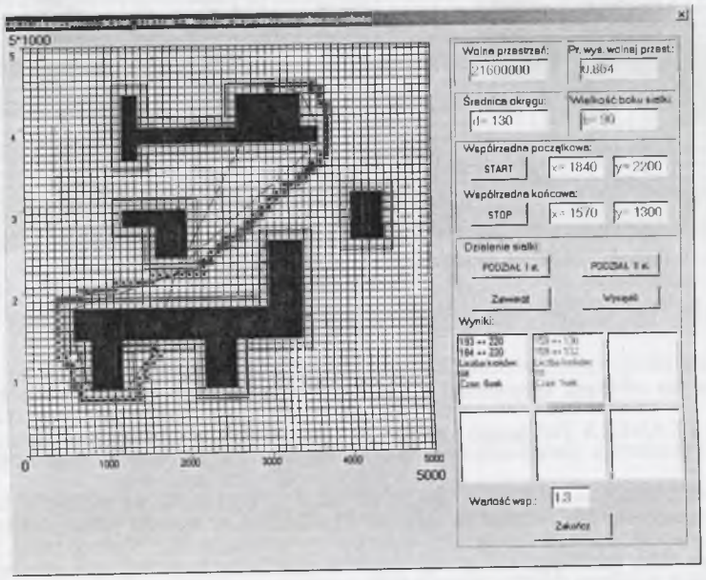
Rys. 6.15. Sklejanie ścieżek bezkolizyjnych

Fig. 6.15. Binding of collision - free paths



Rys. 6.16. Omijanie przeszkody drogą sklejania ścieżek – przypadek 1

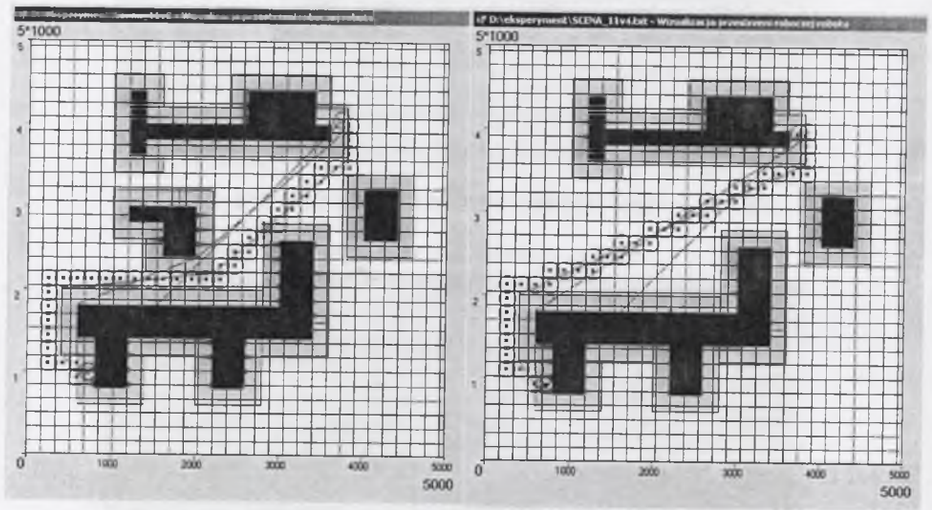
Fig. 6.16. Avoiding an obstacle along the route of paths binding – Case 1



Rys. 6.17. Omijanie przeszkody drogą sklejania ścieżek – przypadek 2

Fig. 6.17. Avoiding an obstacle along the route of paths binding – Case 2

Opracowana metoda pozwala również do zastosowania jej w środowiskach przemysłowych, charakteryzujących się pewną dynamiką układu geometrycznego sceny. Okresowe pojawianie się obiektów w wybranych miejscach sceny może być śledzone nawet prostymi układami sensorów binarnych, a informacja o wykryciu obecności obiektu, lub jego wyjściu z obszaru sceny może być wykorzystana przez PLANER. Zadanie takie w niczym nie zmienia ogólności przedstawionych rozwiązań, co ilustruje rys. 6.18. Oczywiście warunkiem poprawnego radzenia siebie PLANERA w takiej sytuacji jest konieczność zdefiniowania geometrycznego obiektu (co do postaci geometrycznej i miejsca na scenie), gdyż PLANER musi określić dla niego strefę V_{OB} i odpowiednie wagi wynikające z modyfikacji przestrzeni, spowodowanej obecnością obiektu lub jej brakiem.

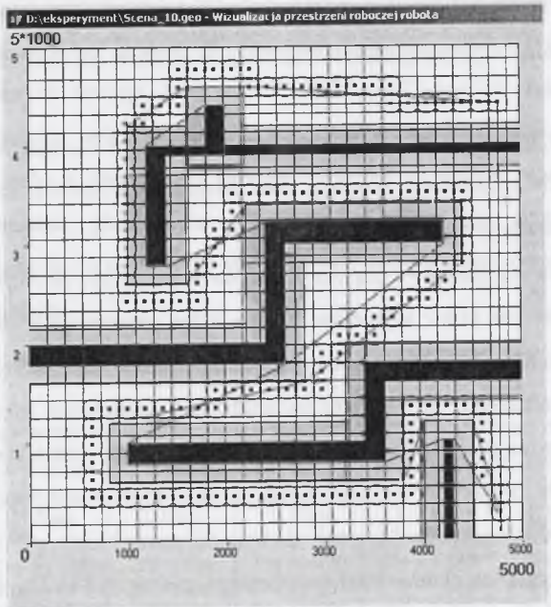


Rys. 6.18. Reakcja PLANERA na procesy dynamiczne zachodzące w scenie robota

Fig. 6.18. The PLANNER reacts to dynamic processes occurring in the operation scene of the robot

Końcowym efektem sprawności i ogólności opracowanej metody może być ilustracja sprawności PLANERA radzącego sobie w przypadkowo skonfigurowanym środowisku (rys. 6.19).

Z zaprezentowanych rezultatów działań PLANERA w sposób jednoznaczny widać, że proces planowania przebiega ściśle wg rygorów określonych strategiami lokalnymi ψ_1 i ψ_2. Dodatkowo, proces dokonywania kolejnych podziałów na poziomach wyższych niż określony poziom startowy występuje stosunkowo rzadko. Można stwierdzić, że miejsca, w których podziały takie mogłyby być konieczne, czyli miejsca leżące w bezpośredniej bliskości



Rys. 6.19. Poszukiwanie optymalnej ścieżki w złożonym środowisku geometrycznym sceny robota

Fig. 6.19. Search for the optimal path in a complex geometrical environment of the operation scene of the robot

obiektów sceny – przeszkód, są dość skutecznie wyeliminowane poprzez zastosowanie strefy V_{OB} i odpowiedniego systemu wag. Rozszerzaniu procesu podziałów „nie sprzyja" również przyjęty warunek prostopadłego zbliżania się i oddalania od przeszkody, gdyż dzięki temu jest on zawężony do koniecznego minimum. W efekcie skutecznie wyeliminowano proces przypadkowego błądzenia, a wartości określonych prawdopodobieństw osiągnięcia wytypowanego stanu (rastra) (5.34) i przejścia do niego (5.35), kształtujące wysokość uzyskiwanej oceny (5.37) i (5.38), zwiększają dodatkowo dążność do możliwie jak najszybszego osiągnięcia zadanego celu. Spostrzeżenia te potwierdzają wyniki szczegółowe, uzyskiwane każdorazowo w procesie planowania.

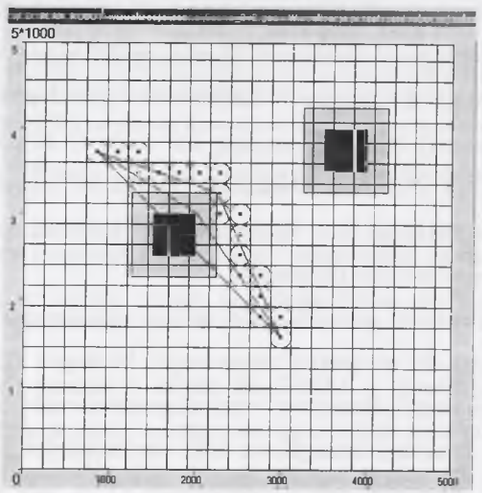
Ze względu na konieczność przetestowania metody opracowany symulator PLANERA typuje ścieżki bezkolizyjne dla wszystkich znalezionych łamanych *TL*, a następnie wskazuje tę najkrótszą, która spełnia optymalnościowy warunek poszukiwania, czyli ścieżkę najkrótszą. Generowane w procesie symulacji szczegółowe dane współrzędnych punktów węzłowych P_{wi} poszukiwanych ścieżek zawierają ciąg współrzędnych (X,Y) tych punktów (tablica 6.1), uzyskanych w odniesieniu do znalezionych łamanych ze zbioru *TL* (pkt 5.6.1.2) w oparciu o ψ_2 i ogólnej strategii maksymalizacji uzyskiwanych ocen (6.39). Ze względu na kryterium

minimalizacji liczby kroków, jakie przynależy ogólnej strategii maksymalizacji zdobywanych ocen (5.2), czyli $i \rightarrow min$ i kryterium najkrótszej drogi przyjęte w ψ_2, działanie PLANERA jest trzyetapowe:

- w pierwszym, typowana jest najkrótsza łamana za zbioru *TL*, zgodnie z ψ_2,
- w drugim, dla znalezionej łamanej, poszukiwane jest rozwiązanie minimalizujące liczbę zrealizowanych kroków t_i, czyli $i \rightarrow min$,
- w trzecim, dla przypadków, dla których może się zdarzyć, że rozwiązań o tej samej liczbie kroków jest kilka; wybierane jest to, które gwarantuje rozwiązanie najlepsze ze względu na długoterminową strategię maksymalizacji zdobywanych nagród.

Mechanizm taki jest niezbędny ze względu na możliwość wystąpienia lokalnych podziałów, które spowalniają proces planowania, są źródłem dodatkowych podziałów i niekoniecznie prowadzą dostatecznie szybko do wyznaczonego miejsca docelowego, jak również wpływu strefy V_{OB} na odległość poszukiwanej ścieżki od omijanej przeszkody.

Dla prostego przypadku planowania, określonego jak na rys. 6.20, dane szczegółowe zawierające współrzędne *x* i *y* (w zapisie: *x++y*) ujęto w tablicy 6.1 (na symulatorze – np. rys. 6.15, 6.16, 6.17 – zapisywane są w oknach *Wyniki*, w prawym dolnym rogu ekranu). Wyniki te uzyskano dla danych: $b=250$ *mm*, $C_k=0.98$ i $b^s=240$ *mm* (wartości *x* i *y* zawarte w tablicy 6.1 wymagają pomnożenia przez *10*, by odpowiadały warunkom sceny o wymiarach *5000x5000 mm*).



Rys. 6.20. Przykładowy układ generujący drzewo decyzyjne *V*

Fig. 6.20. Exemplary system generating decision tree *V*

Uzyskane wyniki poszukiwania przez PLANER rozwiązania najlepszego określają zbiór rozwiązań dopuszczalnych. Wyniki te naniesiono na plan zadania i dla poszczególnych typowanych rozwiązań opisują je rys. 6.21, 6.22, 6.23 i 6.24. Jak widać, elementami wskazującymi przebieg planowania są naroża widoczne z położenia początkowego (*x2760,y1560*). Każde z naroży, do którego zmierza PLANER, wyznacza kolejny *Etap* prowadzonego procesu planowania. Wszystkie rozwiązania wynikają z kombinacji kolejności, typowanych metodą grafu widoczności, naroży napotkanej przeszkody. Uwagę zwracają dwa przypadki:

Tablica 6.1

Dane geometryczne uzyskane z analizy drzewa decyzyjnego *V* dla przykładu z rys. 6.20

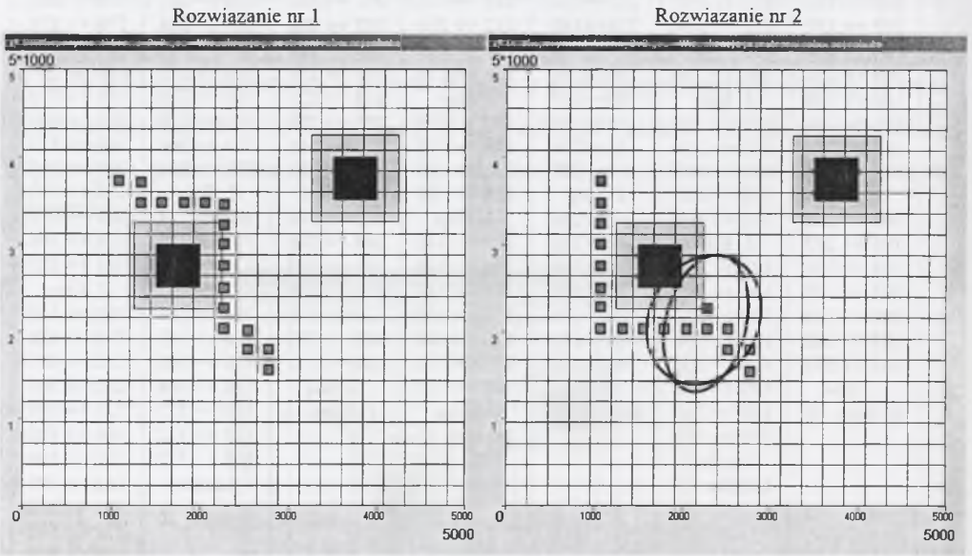
Rozwiązanie nr 1	Rozwiązanie nr 2	Rozwiązanie nr 3	Rozwiązanie nr 4	**Rozwiązanie nr 5**	Rozwiązanie nr 6	Rozwiązanie nr 7
1. Etap*	1. Etap	1. Etap	1. Etap	**1. Etap**	1. Etap	1. Etap
276 ++ 156	276 ++ 156	275++156	276 ++ 156	**276 ++ 156**	276 ++ 156	276 ++ 156
252 ++ 156	276 ++ 180	252++156	276 ++ 180	**276 ++ 180**	276 ++ 180	276 ++ 180
252 ++ 180	252 ++ 180	252++180	252 ++ 180	**276 ++ 204**	276 ++ 204	276 ++ 204
228 ++ 180	252 ++ 204	228++180	252 ++ 204	**252 ++ 204**	300 ++ 204	276 ++ 228
204 ++ 180	228 ++ 204	204++180	228 ++ 204	**252 ++ 228**	300 ++ 228	300 ++ 228
204 ++ 204	228 ++ 228	204++204	228 ++ 228	**252 ++ 252**	300 ++ 252	300 ++ 252
180 ++ 204	2. Etap	180++204	2. Etap	**252 ++ 276**	300 ++ 276	300 ++ 276
156 ++ 204	228 ++ 204	156++204	228 ++ 252	**228 ++ 276**	324 ++ 276	300 ++ 300
132 ++ 204	204 ++ 204	132++204	228 ++ 276	**228 ++ 300**	324 ++ 300	300 ++ 324
108 ++ 204	180 ++ 204	108++204	228 ++ 300	**228 ++ 324**	324 ++ 324	300 ++ 348
2. Etap	156 ++ 204	2. Etap	228 ++ 324	**2. Etap**	2. Etap	324 ++ 348
108 ++ 228	132 ++ 204	108++228	3. Etap	**228 ++ 348**	300 ++ 324	324 ++ 372
108 ++ 252	3. Etap	108++252	228 ++ 348	**204 ++ 348**	276 ++ 324	324 ++ 396
108 ++ 276	108 ++ 204	108++276	204 ++ 348	**180 ++ 348**	276 ++ 348	324 ++ 420
108 ++ 300	108 ++ 228	108++300	180 ++ 348	**156 ++ 348**	252 ++ 348	2. Etap
108 ++ 324	108 ++ 252	108++324	156 ++ 348	**132 ++ 348**	228 ++ 348	300 ++ 420
108 ++ 348	108 ++ 276	108++348	132 ++ 348	**132 ++ 372**	204 ++ 348	276 ++ 420
108 ++ 372	108 ++ 300	108++372	132 ++ 372	**108 ++ 372**	180 ++ 348	252 ++ 420
Liczba	108 ++ 324	Liczba	108 ++ 372	**Liczba**	156 ++ 348	252 ++ 396
kroków: 16	108 ++ 348	kroków: 16	Liczba	**kroków: 16**	156 ++ 372	228 ++ 396
	108 ++ 372		kroków: 16		132 ++ 372	204 ++ 396
	Liczba				108 ++ 372	180 ++ 396
	kroków: 18				Liczba	156 ++ 396
					kroków: 20	156 ++ 372
						132 ++ 372
						108 ++ 372
						Liczba
						kroków: 24

* oznacza odcinek łamanej związany z jednym promieniem widoczności

a) rys. 6.21, *Rozwiązanie nr 2*: w narożu przeszkody zachowanie symulatora wynika z faktu, że w *etapie 1* od strony położenia startowego strefa V_{OB} obiektu umożliwia dotarcie do wyznaczonego naroża. Z kolei strefa V_{OB} blokuje przejście wzdłuż obiektu

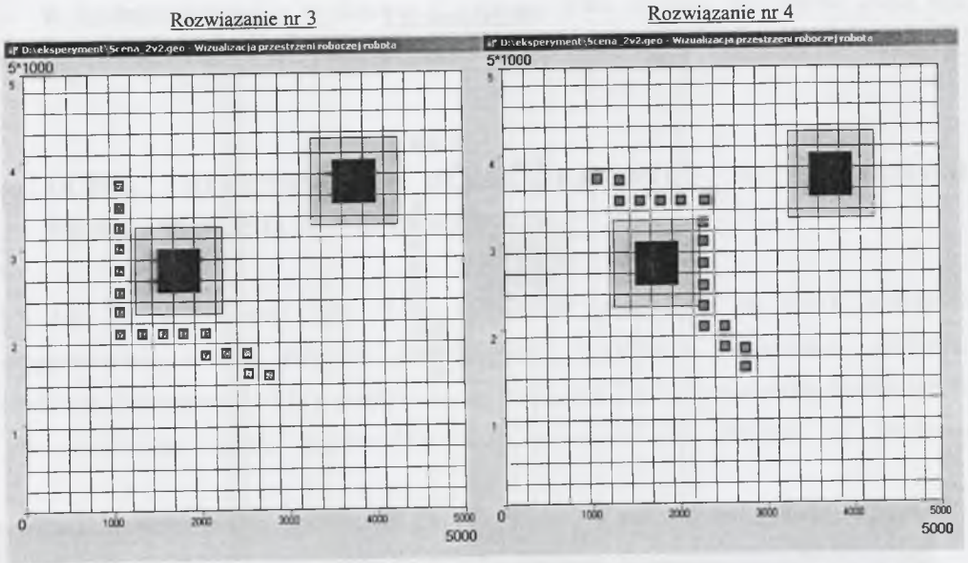
do drugiego naroża w *etapie 2*. Dlatego w narożu pierwszym (miejsce zakreślone) PLANER typuje, jakby dodatkowy, zbędny raster;

b) dla danych zawartych w tablicy 6.1 aż cztery przypadki charakteryzują się taką samą liczbą 16 realizowanych kroków t_i potrzebnych do przejścia z położenia początkowego do końcowego planowanej ścieżki. Spośród nich wybrana, jako najlepsza, została ścieżka odpowiadająca rozwiązaniu nr 5 (tabl. 6.1, rys. 6.23 - rozwiązanie 5). Dzieje się tak dlatego, że ta sama liczba kroków nie odpowiada jeszcze najkrótszej drodze. Długość łamanej *TL* wyznaczona jest względem poszczególnych naroży, zaś obejście przeszkody realizowane jest wg linii prowadzącej, na którą wpływa strefa V_{OB}. Dlatego liczba kroków jest wypadkową zarówno typowanego sposobu obejścia przeszkody (łamana *TL*), jak i bardzo ograniczonego procesu błądzenia PLANERA w ramach korytarza określonego przez C_T (pkt 5.6.1.1).



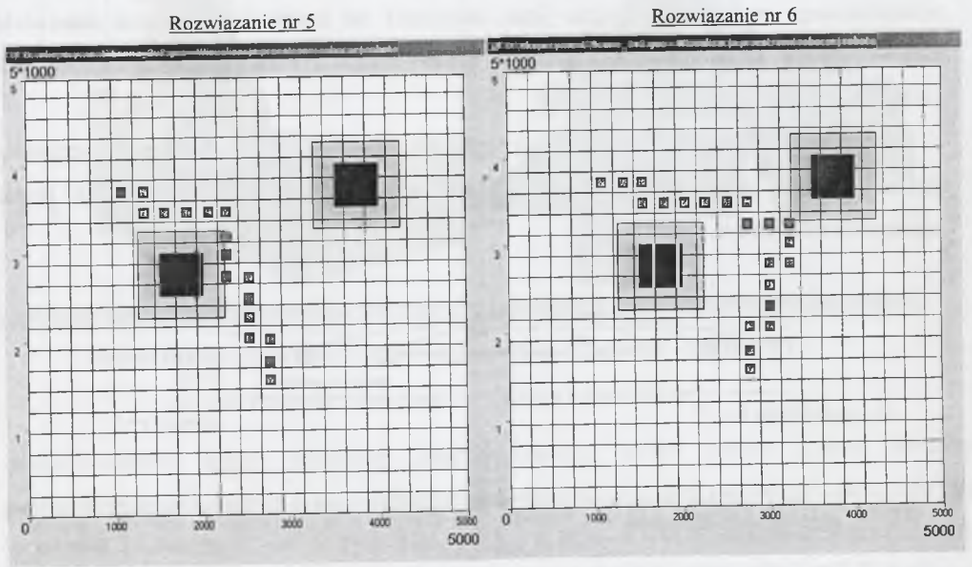
Rys. 6.21. Wyniki szczegółowe symulacji: rozwiązanie 1 i 2

Fig. 6.21. Detailed results of the simulation: Solutions 1 and 2



Rys. 6.22. Wyniki szczegółowe symulacji: rozwiązanie 3 i 4

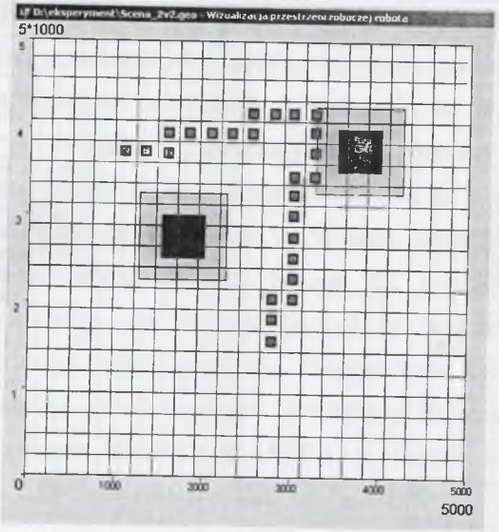
Fig. 6.22. Detailed results of the simulation: Solutions 3 and 4



Rys. 6.23. Wyniki szczegółowe symulacji: rozwiązanie 5 i 6

Fig. 6.23. Detailed results of the simulation: Solutions 5 and 6

Rozwiazanie nr 7

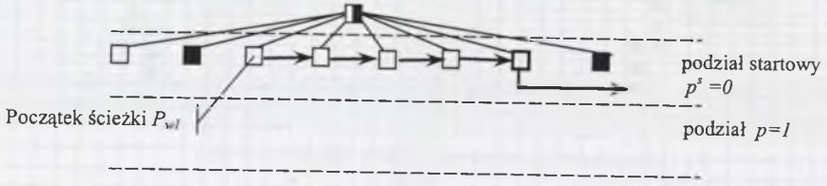


Rys. 6.24. Wyniki szczegółowe symulacji: rozwiązanie 7

Fig. 6.24. Detailed results of the simulation: Solution 7

Uwagę zwraca również podany za rozwiązaniem optymalnym całkowity czas uzyskania poszukiwanego rozwiązania. Mimo jego zależności od szybkości procesora stanowiska komputerowego, na którym został przeprowadzony proces symulacji, jest on zadowalająco krótki.

Z analizy uzyskanych danych wynika również, że graf V, reprezentujący przestrzeń poszukiwań, w której planowana jest poszukiwana ścieżka bezkolizyjna, ma postać zaprezentowaną na rys. 6.25.



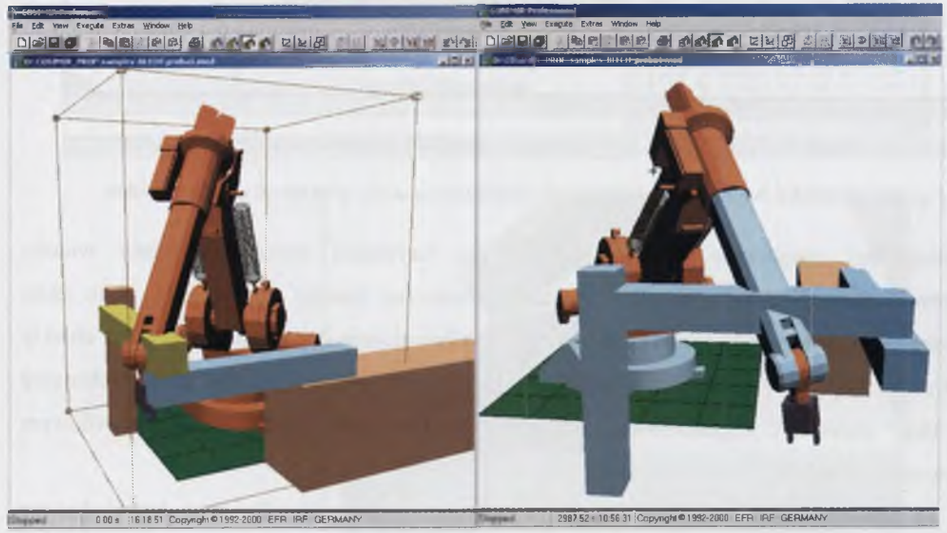
Rys. 6.25. Drzewo V podziału przestrzeni dla przypadku z rys. 6.19. Linią grubą zaznaczono możliwy przebieg planowanej ścieżki robota

Fig. 6.25. Tree V of space division for the example shown in Fig. 6.19 above. Possible course of the planned path of the robot is marked by the line in bold

W ogólnym przypadku w drzewie V lokalnie, tylko wybrane wierzchołki mogą być rozwijane na poziomach wyższych (rys. 5.8).

6.1. OCENA KOLIZYJNOŚCI DZIAŁAŃ ROBOTA W SYSTEMACH PROGRAMOWANIA OFF–LINE ROBOTÓW

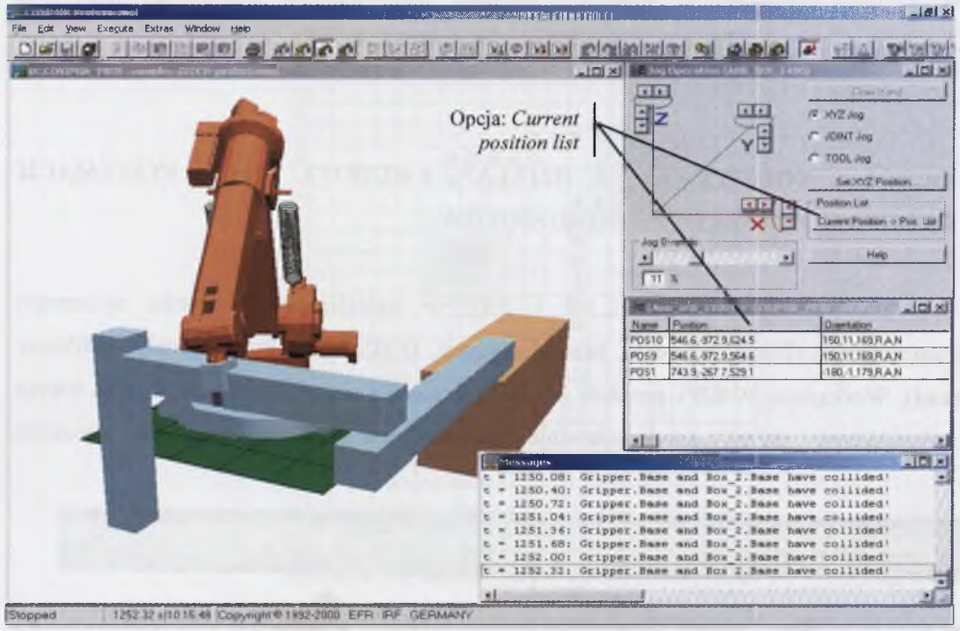
Jak już wspomniano (pkt. 2.1.3 i 2.2), w istniejących na rynku systemach programowania off-line robotów przemysłowych (COSIMIR Professional, eMPower (Robcad), Workspace, IGRIP), problem oceny kolizyjności programowanych działań robota jest rozwiązany raczej powierzchownie. Koncentruje się głównie na założeniu



Rys. 6.26. COSIMIR Professional v.5. Kontrola kolizyjności – przykład 1

Fig. 6.26. COSIMIR Professional v.5. Collision control – Case 1

przeprowadzenia oceny wizualnej programowanego ruchu robota. Celem bardziej szczegółowego omówienia tego mechanizmu zilustrowano go przykładem działania systemu COSIMIR Professional, ver. 5. (rys. 6.26, 6.27. i 6.28) – z wykorzystaniem opcji: *Collision detection*. Generalną zasadą jest możliwość wzajemnego kolidowania wszystkich obiektów scen z robotem. Dopuszczenie do zjawiska wzajemnego przenikania się obiektów, a co za tym idzie – możliwość „zanurzania" się robota w obsługiwanym obiekcie sceny, jest mechanizmem formalnie akceptowanym przez system. Jedynym elementem oceny zaistniałej kolizji

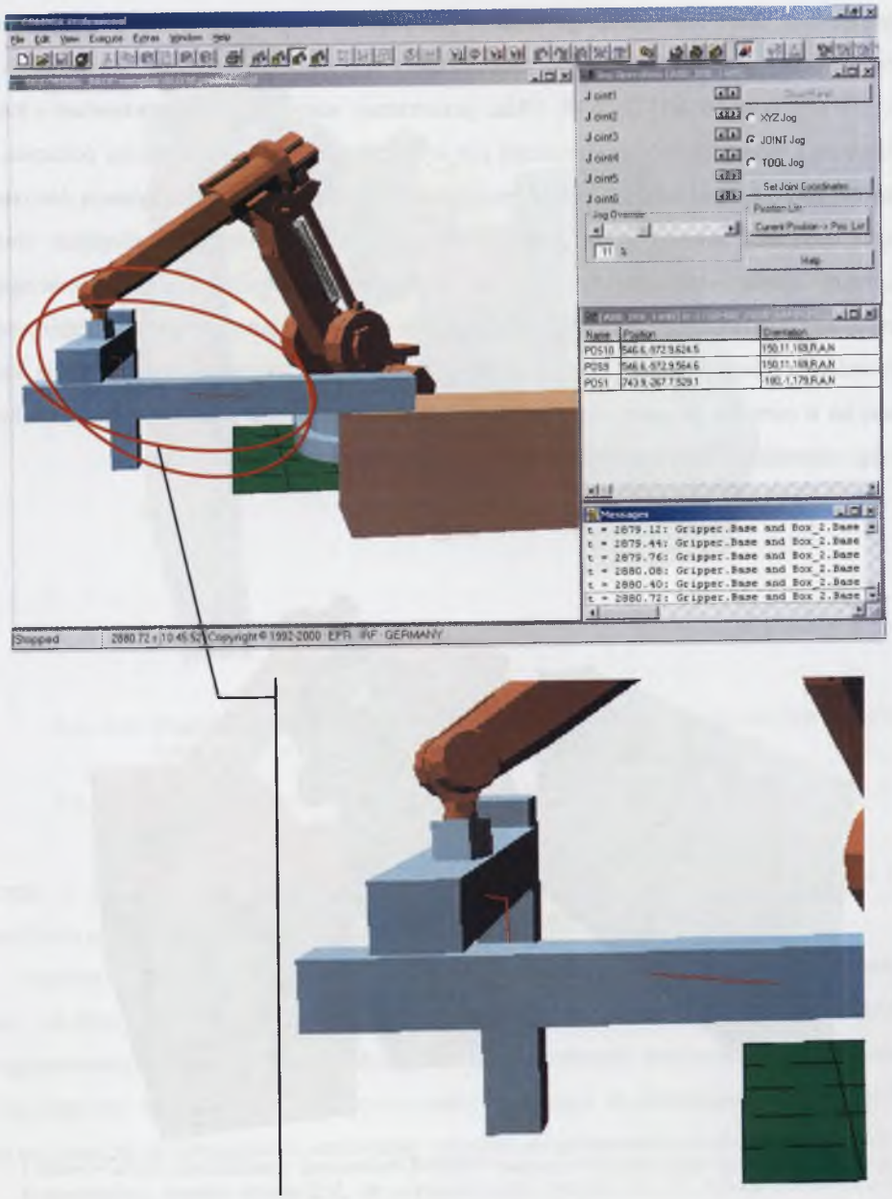


Rys. 6.27. COSIMIR Professional, v.5. Wykrywanie kolizyjności obiektów sceny i chwytaka robota

Fig. 6.27. COSIMIR Professional v.5. Detection of collision capability of technical objects and robot

(uruchamianej opcjonalnie przez operatora) jest uzyskanie wizualnej zmiany widoku (przebarwienia elementów będących w stanie wzajemnej kolizji). Towarzyszy temu okno informacji (Messages) z komentarzem (rys. 6.27) informującym operatora o tym, które obiekty są w stanie kolizji i w jakim czasie ona zaszła. Dodatkowo, moment wejścia jednego elementu „w drugi" może być sygnalizowany krótkim sygnałem dźwiękowym (*beep*) i chwilowym zatrzymaniem robota.

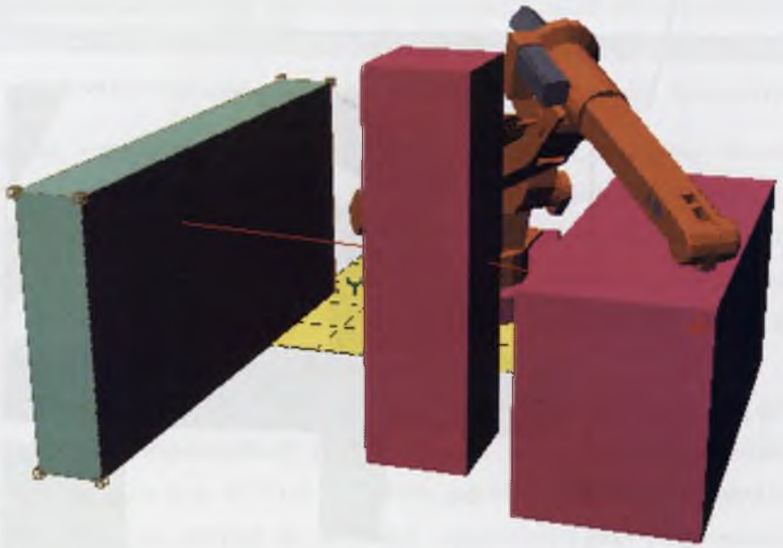
To zatrzymanie jest ważne w celu wychwycenia pozycji robota, w której kolizja zachodzi. Wartości współrzędnych początku kolizji można odczytać w oknie *Current Position: Position List* (rys. 6.27 i 6.28). Zatem jedynie te informacje o zaistniałej kolizji robota z przeszkodą są możliwe do uzyskania (opcjonalnie!) przy programowaniu robota metodą off-line. Efekt zaprogramowania robota bez uwzględnienia zachodzącej kolizji pokazano na rys. 6.28. Jest nim zaprogramowana trajektoria (linia czerwona) przechodząca przez obiekty sceny. Jest rzeczą oczywistą, że sposoby unikania kolizji, czyli poszukiwanie dróg obejścia napotkanej przeszkody, zależą wyłącznie od subiektywnej interpretacji sytuacji kolizyjnej przez operatora-programistę robota.



Rys. 6.28. COSIMIR Professional, v.5. Kolizyjność trajektorii robota. Programowana trajektoria zaznaczona jest kolorem czerwonym

Fig. 6.28. COSIMIR Professional v.5. Collision capability of robot trajectory. The programmed trajectory is indicated in red

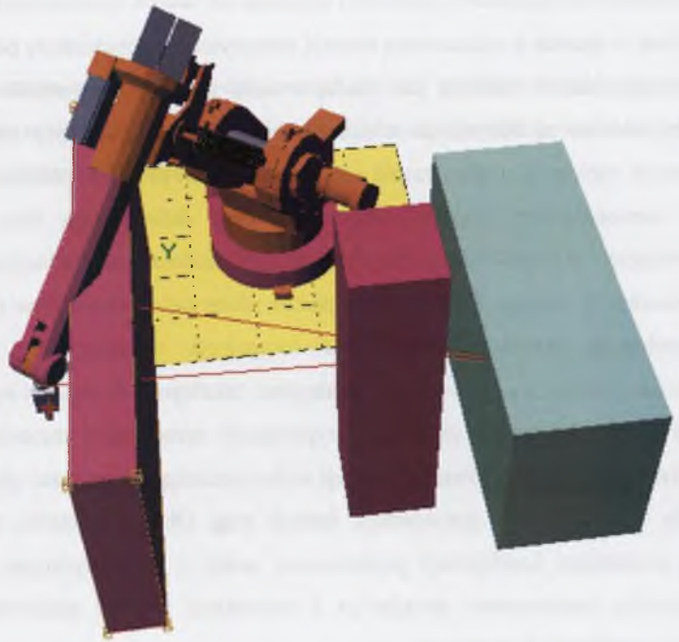
W celu bezpośredniego porównania rezultatów działania systemu COSIMIR Professional i programu Wiz_SRM rozwiązującego zadanie planowania w oparciu o przedstawioną metodę opracowano prosty przykład symulacyjny odpowiadający przypadkowi planowania pokazanemu na rys. 6.12 i 6.13 oraz od 6.15 do 6.18. Układ przestrzenny sceny robota skomponowano z trzech brył (obiektów), pomiędzy którymi konieczne jest wygenerowanie ścieżki robota. Jak pokazano na rysunkach 6.12 i 6.13 oraz od 6.15 do 6.18, program Wiz_SRM radzi sobie z tą sytuacją doskonale, generując optymalną ścieżkę, która przebiega pomiędzy tymi obiektami w odległości równej przynajmniej strefie oddziaływania obiektu. COSSIMIR Professional mimo znaczącego zaawansowania programistycznego i graficznego nie daje możliwości automatycznego i bezpiecznego przeprowadzenia robota pomiędzy nimi (rys. 6.29). Obowiązek ten spoczywa na programiście, który podobnie jak w metodzie programowania przez uczenie sam musi wybrać ścieżkę obejścia przeszkody dobierając odpowiednio kolejne punkty węzłowe programowanej ścieżki (rys. 6.30).



Rys. 6.29. Realizacja ruchu robota w systemie COSIMIR Professional. Kontrola kolizyjności – widok 1

Fig. 6.29. Robot motion implemented COSIMIR Professional v.5. Collision control – View 1

Jak widać, mechanizm badania kolizyjności programowanego ruchu robota pokazany na rys. 6.26, 6.27 i 6.28 nie pozwala na uniknięcie kolizji. Jedynym sygnalizatorem zajścia takiego zdarzenia jest zmiana koloru brył będących w stanie kolizji. Jednak COSIMIR umożliwia wizualną ocenę kolizji wszystkich obiektów znajdujących się w przestrzeni robota,



Rys. 6.30. Programowanie ruchu robota w systemie COSIMIR Professional. Kontrola kolizyjności – widok 2

Fig. 6.30. Programming the robot motion in COSIMIR Professional v.5. Collision control – View 2

łącznie z ogniwami jego łańcucha kinematycznego. Tego zaś z założenia (pkt 4.1) nie umożliwia opracowana metoda na obecnym etapie rozwoju.

Przedstawiony problem można podsumować następująco: systemy programowania off-line robotów przemysłowych nie są wyposażone w żadne mechanizmy planowania programowanej trajektorii robota i unikania kolizji z obiektami stanowiącymi wyposażenie technologiczne sceny robota. Mechanizm reakcji operatora na dostrzeganą sytuację kolizyjną jest taki sam jak w powszechnie stosowanej metodzie programowania robota przez uczenie.

Reasumując, można stwierdzić, że wykorzystanie prostej co do założeń i efektywnej co do skuteczności i szybkości metody oceny kolizyjności i planowania ścieżki przejścia robota do wskazanego miejsca w przestrzeni byłoby doskonałym narzędziem uzupełniającym oprogramowanie systemowe metod programowania off-line robotów. Warunki takie spełnia przedstawiona w pracy metoda planowania bezkolizyjnych ścieżek robotów manipulacyjnych.

6.3. WNIOSKI

Przeprowadzone w oparciu o opracowaną metodę eksperymenty potwierdziły postawione w pracy główne założenia, iż możliwe jest zaadaptowanie wybranych elementów technik planowania ruchu robotów mobilnych do obszaru robotyki stacjonarnej, bez szkody dla jakości uzyskiwanych rozwiązań. Ograniczenie przeprowadzonych procesów adaptacji metod uczenia się ze wzmocnieniem i grafu widoczności przy definiowaniu funkcji ocen typowanych rozwiązań w środowisku trójwymiarowym robota manipulacyjnego nie pogorszyło sprawności tej funkcji. Dodatkowo, oparcie zdobywanej nagrody na funkcjach składowych określonych prawdopodobieństwami osiągnięcia wyznaczonych punktów węzłowych pozwoliło, łącznie z opracowanymi strategiami lokalnymi, skutecznie ograniczyć głębokość uzyskiwanego drzewa decyzyjnego. Oryginalność opracowanej metody, oprócz wymienionych elementów, wynika również z relacji wykorzystanego w procesie planowania poziomu podziału oktalnego oraz opracowanej funkcji wag. Oba te czynniki, związane jednoznacznie z warunkami konfiguracji przestrzennej sceny i geometrycznymi procesu manipulacji, pozwalają każdorazowo związać je z warunkami procesu planowania, bez szkody dla jakości uzyskiwanych rozwiązań.

Wymiernym i pożądanym efektem tego stanu jest wyeliminowanie stosowanych dotychczas, w zadaniach rastrowej analizy przestrzeni robota, metod przeszukiwania heurystycznego, które w istotny sposób wydłużają czas pozyskiwania rozwiązania dopuszczalnego. Elementami, które pozwoliły wyeliminować konieczność wykorzystania heurez, są zaproponowana szczegółowość podziału oktalnego na poziomie startowym i strategie lokalne. Składniki te, łącznie ze wspomnianą już funkcją nagród, sprawiły, że działanie PLANERA jest ściśle zdeterminowane, co oznacza, że zarówno zjawisko błądzenia, jak również jego skutki w postaci występowania pętli i minimów lokalnych nie występują. Zalety opracowanej metody dobrze ilustrują uzyskane wyniki. Równocześnie uwagę zwraca fakt, że metoda ta pozwala planować ścieżki robota nawet w środowisku o „swobodnie i ciasno" rozmieszczonych przeszkodach. Jest to cecha, która wynika wprost z zastosowanych technik robotyki mobilnej. Oznacza to, że przeprowadzone eksperymenty potwierdziły również, że metoda stwarza dogodne warunki do podjęcia prób zastosowania jej w systemach programowania robotów manipulacyjnych, charakteryzujących się dużą dynamiką.

Porównując uzyskane wyniki pracy programu Wiz-SRM i systemu COSIMIR Professional można stwierdzić, że opracowana metoda może stanowić, zgodnie z założeniami, uzupełnienie stosowanych rozwiązań w graficznych systemach programowania off-line robotów, dzięki którym można by rozwiązać problem automatycznego generowania najkrótszej (optymalnej) ścieżki do zadanego położenia, zwalniając tym samym programistę z tego zadania.

Można zatem stwierdzić, że opracowana metoda spełnia wszystkie przyjęte założenia i kryteria. Tym samym uzyskane wyniki pozwoliły na osiągnięcie postawionego w pracy celu i dowiedzenie słuszności sformułowanej tezy.

7. Podsumowanie

Zaprezentowana praca powstała w efekcie wieloletniego zainteresowania autora zagadnieniami związanymi z automatyzacją i robotyzacją procesów technologicznych, w szczególności takimi jak systemy CAD/CAM i programowanie robotów przemysłowych. Doświadczenia, poparte również przemysłową praktyką zawodową, wskazują jednoznacznie, że w sferze szeroko pojętej automatyzacji produkcji robotyzacja procesów zajmuje coraz bardziej znaczące miejsce. Wynika to z prostego faktu, jakim jest dążenie do wyeliminowania najbardziej zawodnego elementu produkcji, którym jest udział nisko kwalifikowanych czynności roboczych na stanowiskach produkcyjnych. Potrzebie tej sprzyja proces, czasami bardzo dotkliwy społecznie, jakim jest szeroko pojęta robotyzacja procesów obsługi, a nawet czynności technologicznych. Niestety, procesowi temu nie towarzyszy równie szybki i efektywny rozwój technik pozwalających na efektywne wdrażanie robotów przemysłowych do produkcji. Dzieje się tak mimo gwałtownego rozwoju robotyki, stymulowanego osiągnięciami głównie automatyki, informatyki i mechaniki. Dodatkowo, szybki rozwój bardziej atrakcyjnej robotyki mobilnej sprawia, że zainteresowanie badaczy robotami stacjonarnymi i manipulacyjnymi wydaje się mniejsze. Skutkiem takiego stanu jest m. in.:

- skierowanie uwagi na procesy sterowania robotami stacjonarnymi w celu poprawienia ich jakości,
- dominacja w dalszym ciągu programowania metodą samouczenia. Nowoczesne techniki komputerowe oparte na programowaniu graficznym nie są dość powszechne.

Rozwój poziomu oprogramowania układów sterowania robotów przemysłowych prowadzi do poprawy jakości programowania samouczącego, głównie za sprawą pojawiających się coraz szerzej bibliotek typowych procedur programowych (np. układy COMAU C3G*Plus* i C4G*Plus*, czy ASEA S4C*plus*) oraz ogólnych języków programowania przeznaczonych dla pewnego typoszeregu i klasy robotów manipulacyjnych (np. PDL2 i jego bardziej przyjazna

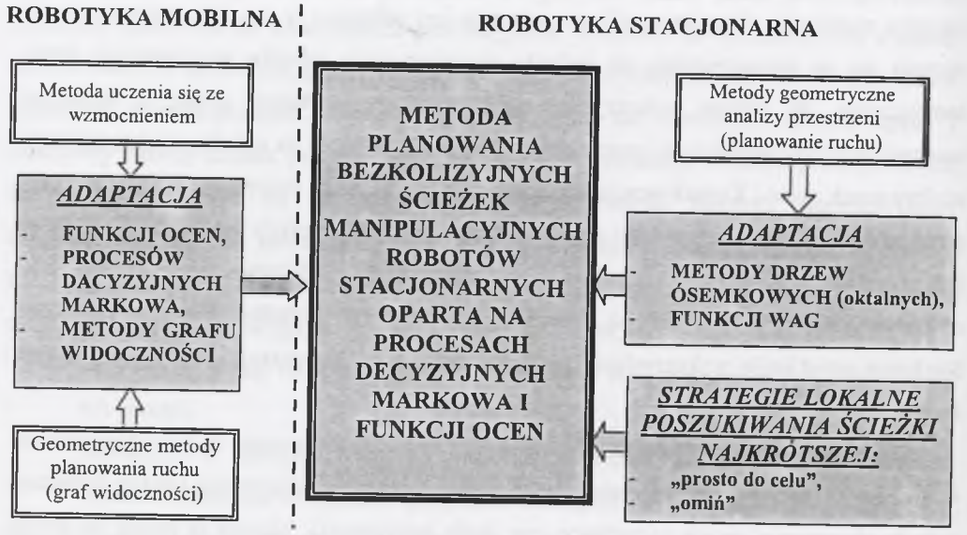
wersja EZ-PDL2, firmy COMAU). Dzięki takim rozwiązaniom możliwe jest tworzenie sparametryzowanych procedur programowych robota, zawierających zarówno definicje elementów otoczenia, biblioteki obiektów sceny, narzędzi robota, chwytaków, jak i sparametryzowane trajektorie robota. Podobne mechanizmy zawierają systemy programowania graficznego off-line robotów (np. COSIMIR, eMPower czy IGRIP). We wszystkich tych systemach brakuje jednak elementu oceny jakościowej programowanych działań robota, począwszy od sekwencji procesu obsługi (cyklogram), a skończywszy na definiowanych trasach przejazdu robota w środowisku technologicznym. Jak już wspomniano (pkt 2), problem kolizyjności projektowanych dróg robota ogranicza się tylko do sprawdzenia metodą porównania współrzędnych, czy wyznaczone położenia związane z obejściem przeszkody lub miejscem obsługi nie pokrywają się ze znajdującymi się w otoczeniu robota obiektami. Tor przejazdu robota pozostaje zaś poza procesem kontroli i weryfikacji. Z tych powodów podjęcie przez autora próby opracowania skutecznej i efektywnej metody planowania bezkolizyjnych ścieżek robotów stacjonarnych i manipulacyjnych było zasadne.

W opracowanej i przedstawionej w pracy metodzie wykorzystano wybrane elementy wydajnych obliczeniowo i skutecznych technik planowania bezkolizyjnych trajektorii robotów mobilnych. U podstaw takiego podejścia jest wskazana w pkt. 5.1 analogia metody uczenia się ze wzmocnieniem do techniki programowania robotów stacjonarnych drogą samouczenia. W efekcie wykorzystano w opracowanej metodzie prosty i skuteczny mechanizm oceny postępowań opracowanego PLANERA, oparty na zasadzie maksymalizacji zdobywanych ocen. Konsekwencją tego „zapożyczenia" była konieczność wykorzystania procesu decyzyjnego Markowa do estymacji funkcji ocen osiąganych przez PLANER, który jest modelem matematycznym uczenia się ze wzmocnieniem. Równocześnie oparcie ocen zdobywanych przez PLANER na funkcjach warunkowego prawdopodobieństwa procesu Markowa umożliwiło wykorzystanie zasad sterowania stochastycznego w rozwiązywanym zadaniu planowania.

Kolejnym „zapożyczeniem" z robotyki mobilnej jest zaadaptowana (z powodzeniem) do warunków trójwymiarowej przestrzeni planowania robota manipulacyjnego, bardzo skuteczna metoda planowania, oparta na technice tzw. grafu widoczności. Metoda ta należy do grupy wydajnych obliczeniowo metod planowania ścieżek robotów mobilnych, które mogą być wykorzystywane w technikach sterowania on-line [43]. Graf widoczności rozszerzony do warunków przestrzeni 3D stał się podstawą opracowanych dwóch strategii lokalnych.

Te elementy są oryginalnym rozwiązaniem wskazującym na nową jakość, jaką udało się osiągnąć w opracowanej metodzie, a jest nią wysoka wydajność czasowa prowadzonych analiz przestrzeni. Relacje te, w oparciu o tablicę 4.1, pokazano na rysunku 7.1.

		METODY ANALIZY PRZESTRZENI ROBOTA		
		Topologiczne	GEOMETRYCZNE (GRAFOWE)	Analityczno-geometryczne
METODY PLANOWANIA RUCHU ROBOTÓW	STACJONARNYCH	—	METODA PLANOWANIA BEZKOLIZYJNYCH ŚCIEŻEK ROBOTÓW STACJONARNYCH OPARTA NA PROCESACH MARKOWA I FUNKCJI OCEN	[38,43,110,132,220]
	MOBILNYCH	[15,36,38,61,77, 86,111,113,119, 205,135]	[43,111,132,149]	[43,56,69,111]



Rys. 7.1. Oryginalne elementy opracowanej metody planowania bezkolizyjnych ścieżek robotów stacjonarnych. Kolorem szarym zaznaczono oryginalne rozwiązania opracowanej metody

Fig. 7.1. Innovative elements in the discussed method of planning the collision-free paths of stationary robots. The Author's original solutions within the framework of the method are marked in grey

Spośród cech, jakimi charakteryzuje się przedstawiona w pracy metoda, dwie wydają się - zdaniem autora - najbardziej istotne. Są to:

- duża prostota zaprezentowanej metody,
- efektywność czasowa.

Prostota metody wynika z przejrzystego mechanizmu kwalifikacji rozwiązań proponowanych przez PLANER, działający w oparciu o nią. Jedną z typowych cech technologów-programistów robotów jest obawa przed skutkami, jakie mogą wywołać w zagęszczonym przeszkodami środowisku programowane ruchy robota. Zaproponowane rozwiązania (czego wyrazem jest przedstawiony system wag) są zbieżne z intuicyjnymi odczuciami, jakie towarzyszą oczekiwanym, bezpiecznym ruchom robota: ruch kreowany jest w bezpiecznej odległości od każdego obiektu, każdy raster wielkością zbliżony jest do przedmiotu manipulowanego (pojęcie przedmiotu manipulowanego może obejmować zarówno sam przedmiot, jak i chwytak), w miejscach związanych z ciasnymi przesmykami między obiektami sceny rastrowanie przestrzeni jest odpowiednio większe. Stąd też metoda nie zaskakuje proponowanymi rozwiązaniami, natomiast gwarantuje typowanie rozwiązań najlepszych ze względu na kryterium najkrótszej drogi.

Efektywność metody wynika z szybkości uzyskiwanych rozwiązań. Zastosowanie strategii lokalnych w sposób skuteczny ogranicza wielkość typowej dla metod rastrowych przestrzeni poszukiwań rozwiązań dopuszczalnych. Dzięki zastosowaniu bardzo szybkiej metody grafu widoczności, która - jak już wspomniano - dla robotów mobilnych może być wykorzystana do planowania ich ścieżek w czasie rzeczywistym, kierunek poszukiwania jest jednoznacznie zdeterminowany wyznaczoną prostą prowadzącą. Zastosowane podwójne kryterium odległości: od wyznaczonego celu poszukiwanej ścieżki i od prostej prowadzącej, łącznie z procesem sterowania stochastycznego, pozwala wyeliminować błądzenie PLANERA w przestrzeni, a przez to takie niepożądane zjawiska, jak minima lokalne czy pętle. Dzięki takiemu rozwiązaniu PLANER koncentruje swoje działania praktycznie na jednym pierwotnym (startowym) poziomie podziału rastrowego przestrzeni, co sprawia, że tylko w sporadycznych przypadkach konieczne jest dokonywanie podziałów na wyższych poziomach. Elementy te praktycznie eliminują konieczność zastosowania heurez do przeszukiwania tworzonego w procesie planowania drzewa decyzyjnego, reprezentującego przestrzeń działania robota i dzięki temu proces selekcji rozwiązania najlepszego jest prosty i szybki. Określenie szybkości dokonywanych analiz zgodnie z przytoczonym w pracy wzorem Stirlinga (pkt 3.2.2.1.1) jest trudne. Jednak z przeprowadzonych symulacji wynika, że rozwiązanie poprawne jest możliwe do uzyskania w czasie od kilku do kilkudziesięciu

sekund, zależnie od złożoności geometrycznej przestrzeni sceny (na przestrzeni kilkudziesięciu kroków). Zatem można uznać, że opracowana metoda spełniła związane z nią oczekiwania (pkt 4). Być może również w stosunku do robotów manipulacyjnych możliwe będzie zastosowanie tej metody do programowania on-line.

Cechy te sprawiają, że również proces jej oprogramowania w celu wykorzystania praktycznego w technice programowania off-line nie nastręcza większych trudności. Forma generowanych danych geometrycznych punktów węzłowych znajdowanych ścieżek pozwala na ich transmisję i wykorzystanie w systemach graficznej wizualizacji, np. w systemach typu CAD/CAM. Prace nad ich wykorzystaniem w systemie CATIA są prowadzone.

Oczywiście, opracowana metoda, w formie zaproponowanej w pracy, nie uwzględnia wielu ważnych aspektów, niezbędnych do jej bezpośredniego wykorzystania w zaawansowanych systemach programowania off-line robotów. Tematami, które powinny być rozwiązane w pierwszej kolejności, są:

- wygładzenie uzyskiwanych ścieżek, którego stopień może być również traktowany jako jedno z kryteriów optymalizacyjnych [45,46,69,75,124,217],
- parametryzacja czasem, a przez to zastąpienie wyznaczonej ścieżki odpowiadającym mu równaniem trajektorii (funkcją przejścia) robota,
- uwzględnienie w planowaniu procesów dynamiki sceny robota i wykorzystanie złożonych układów sensorowych, pozwalających na ich rozpoznanie,
- zastosowanie zaawansowanych metod CAD-owskich do definiowania sceny.

Zagadnienia te wyznaczają dalsze szczegółowe kierunki badań nad wykorzystaniem opracowanej metody do programowania robotów przemysłowych.

Przedstawione i omówione relacje pomiędzy robotyką mobilną i stacjonarną, których efektem praktycznym jest przedstawiona metoda, skłaniają do wyciągnięcia pewnych ogólnych wniosków, które mogą być podstawą szerszego procesu, jakim w robotyce może być korzystanie z osiągnięć robotyki mobilnej i umiejętne adaptowanie ich przez robotykę stacjonarną. Procesy adaptacyjne powinny prowadzić do integracji obu gałęzi robotyki, a przez to wyznaczać kierunki dalszych badań. Dla przykładu, jednym z pierwszych elementów takiej adaptacji, który znalazł swoje zastosowanie zarówno w robotyce mobilnej, jak i stacjonarnej, jest metoda drzew czwórkowych stosowana do zagadnień płaskich w robotyce mobilnej [149]. Znalazła ona swój odpowiednik w zadaniach analizy przestrzeni wykorzystywanych w robotyce stacjonarnej, jako metoda drzew ósemkowych (oktalnych) [110]. Jest ona jedną z podstawowych metod w tym obszarze badań.

Cechą metod robotyki mobilnej jest wymagana obligatoryjnie szybkość procedur sterujących ruchem robota, gdyż pracują one często w technice on-line. Specyfika zagadnień robotyki mobilnej wymaga jednak umiejętnego korzystania z jej osiągnięć i stosowanych narzędzi naukowych. Podstawową kwestią, jaka wiąże się z tym problemem, jest pytanie: które z elementów i w jakim zakresie mogą być najlepiej przydatne w zagadnieniach programowania i sterowania ruchem robotów manipulacyjnych. Odpowiedź na nie może być jednym z kluczowych kierunków rozwoju robotyki, zmierzających do integracji tych dwóch ważnych jej działów. Oczywiście, problem adaptacji jest tu kluczowy. Zakres prac integracyjnych i adaptacyjnych może obejmować całe spektrum problemów, zarówno szczegółowych, jak i ogólnych o kluczowym znaczeniu, i dać nowy impuls robotyce stacjonarnej w takich obszarach badań, jak:

- szersze wykorzystanie technik sensorowych, które w robotyce mobilnej są szeroko stosowane do śledzenia otoczenia i umożliwiają korygowanie wyznaczonych (planowanych) i zaprogramowanych tras przejazdu robota zależnie od jego warunków. Szczególnie atrakcyjnym zakresem badań mogą być elementy wizji maszynowej pracujące w układach z innymi sensorami umożliwiającymi rozpoznawanie i pomiar głębi (układy laserowe, ultradźwiękowe). Zagadnienie to było szerzej omawiane w pkt. 3.2.1;
- możliwość budowy szybkich algorytmów programowania robota, pozwalających na wykorzystanie wysoko zaawansowanych logicznie języków poziomu „task" i programistycznie ich kompilatorów, w których podstawową kwestią jest jego programowanie z poziomu realizowanego zadania obsługi, dzięki wykorzystaniu zorientowanych problemowo baz danych i wiedzy. Kompleksowe podejście do tego zagadnienia może zaowocować efektywnymi rozwiązaniami programistycznymi (sformalizowany język) mającymi swoje solidne umocowanie na płaszczyźnie sterowania procesem planowania akcji robota i jego ruchem;
- poszukiwanie szybkich algorytmów sterowania robotem manipulacyjnym opartych na szybkich procedurach numerycznych, które są podstawą techniką rozwiązywania zadania sterowania z poziomu układu sterowania robotem.

W tych obszarach mieszczą się wszystkie wskazane szczegółowe zagadnienia związane z rozwojem omówionej metody planowania bezkolizyjnych ścieżek robotów manipulacyjnych.

Innym problemem, ważnym zarówno dla robotyki stacjonarnej, jak i mobilnej, jest „rozbicie" zadania planowania ruchu robota na dwa obszary, z których jeden związany jest z jego rozwiązywaniem podejmowanym w przestrzeni zadania, a drugi w uogólnionej przestrzeni stanu. Oczywiste jest, że kompleksowo pojmowany problem szybkości

planowania działań robota jest uzależniony od szybkości jego rozwiązania w obu tych obszarach, co jest szczególnie ważne przy programowaniu robota techniką on-line. Stąd też konieczność poszukiwania metod pozwalających na kompleksowe rozwiązanie tego problemu, co w praktyce, przekładając się na proces efektywnego planowania w sferze zadaniowej (programistycznej) i sterowniczej, wydaje się pierwszoplanowym zadaniem. Przykładowym obszarem wykorzystania tych rozwiązań mogą być zadania sterowania grupą współpracujących ze sobą robotów. Dopiero takie podejście pozwoli rozwiązać wszystkie problemy związane z optymalizacją ruchu robota w kategoriach globalnych, co do tej pory nie było możliwe.

Tak więc, konkludując, można stwierdzić, że rozwój współczesnej robotyki, szczególnie stacjonarnej, podobnie jak współczesnych systemów produkcyjnych, leży w obszarze badań prowadzących do pełnej integracji i spójności metod rozwiązujących szeroko pojęty problem programowania i sterowania robotem, czego przykładem może być integracja metod i technik robotyki stacjonarnej i mobilnej.

Literatura

1. Arai Y., Fujii T., Asama H., Kaetsu H., Endo I.: Collision avoidance in multi-robot systems based on multi-layered reinforcement learning. Robotics and Autonomous Systems 29(1999), Elsevier 1999, pp. 21-32.
2. Asada M., Uchibe E., Hosoda K.: Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development. Artificial Intelligence 110(1999), Elsevier 1999, pp. 275-292.
3. Asep R., Achaibou A.K., Mora-Camino F.: Automatic Collision Avoidance Based on Supervised Predictive Controllers. Control Engineering Practice, Vol. 4, No. 8, Pergamon Press 1996, pp. 1169-1175.
4. Autere A.: Hierarchical A* based path planning – a case study. Knowledge-Based Systems 15(2002), Elsevier 2002, pp. 53-66.
5. Awrejcewicz J., Mosdorf R.: Analiza numeryczna wybranych zagadnień dynamiki chaotycznej. WNT, Fundacja „Książka Naukowo-Techniczna", Warszawa 2003.
6. Awrejcewicz J., Wodzicki W.: Podstawy automatyki: teoria i przykłady. Wydawnictwo Politechniki Łódzkiej, Łódź 2001.
7. Awrejcewicz J.: Drgania deterministyczne układów dyskretnych. WNT, Warszawa 1996.
8. Awrejcewicz J.: Matematyczne metody mechaniki. Wydawnictwo Politechniki Łódzkiej, Łódź 1995.
9. Bailey T., Nebot Ed.: Localisation in large-scale environments. Robotics and Autonomous Systems, vol.37 (2001), pp. 261-281, Elsevier 2001.
10. Barczyk J., Igieliński J., Łunarski J.: Układy podawania w systemach automatycznego montażu. Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 1996.

11. Bartlett P.L., Baxter P.: Estimation and Approximation Bounds for Gradient-Based Reinforcement Learning. Journal of Computer and Systems Sciences, 64(2002), Academic Press 2002, pp. 133-150.
12. Bellier C., Laugier C., Mazer E., Troccaz J.: A practical system for planning safe trajectories for manipulator robots. in Mechatronics and Robotics, I. Advances in Design and Manufacturing, edtrs: MecConaill P.A., Drews P., Robrock K.-H., OIS Press, Amsterdam 1991, pp. 153-163.
13. Ben-Arieh D.: Task management in a multi-robot environment. Computer Integrated Manufacturing Systems, Vol. 10, No. 2. Elsevier 1997, pp. 123-131.
14. Blum Ch., Jacob W.: Programming Languages for Industrial Robots. Spronger-Werlag, 1986.
15. Boada B.L., Blanco D., Moreno L.: Symbolic Place Recognition in Voronoi-Based Maps by Using Hidden Markov Models. Journal of Intelligent and Robotic Systems, 39 (2004), pp.173-197. Kluwer Academic Publishers 2004.
16. Bolc L., Cytowski J.: Metody przeszukiwania heurystycznego. t. 1, PWN, Warszawa 1989.
17. Borkowski A., Dubrawski A., Racz J.: Sieci neuronowe typu ART w systemach nawigacyjnych autonomicznych robotów. Materiały V Krajowej Konferencji Robotyki. t.1, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 1996, s. 167-176.
18. Bronsztejn I.N., Siemiendiajew K.A.: Matematyka. Poradnik encyklopedyczny. PWN, Warszawa 1990.
19. Carvera E., Pobil del A.P.: Sensor-based learning for practical planning of fine motion in robotics. Information Sciences. An International Journal. Elsevier, 145 (2002), pp. 147-168.
20. Chatila R., Alami R., Giralt G.: Task-Level Programmable Intervention Autonomous Robots. in Mechatronics and Robotics, I. Advances in Design and Manufacturing, Edts: MacConaill P.A., Drews P., Robrock K.-H., IOS Press, Amsterdam 1991, pp.77-87.
21. Chen K. Tham: Reinforcement learning of multiple tasks using a hierarchical CMAC architectuire. Robotics and Autonomous Systems 15/1995, pp. 247-274, Elsevier.
22. Cheng H., Cheng H.D.: Feasible map algorithm for path planning. Robotics and Autonomous Systems, 17(1996), Elsevier 1996, pp. 149-170.

23. Chlebus E., Cholewa M., Borkowski T.: Proces-oriented product development. Proc. of the CAMT Conference "Modern trends in manufacturing", Wrocław 2002. pp. 89-104.
24. Chlebus E., Cholewa M.: Model danych w technologicznym projektowaniu procesów wytwarzania. Materiały V Konferencji „Komputerowo zintegrowane wytwarzanie", t.1, Zakopane 2002. WNT, Warszawa 2002, s. 168-177.
25. Chlebus E., Cholewa M.: Modele danych w TPP. Materiały V Międzynarodowej konferencji naukowej „Komputerowe wspomaganie prac inżynierskich". t.1. Wrocław 2000. s. 185-192.
26. Chlebus E., Duda J., Knosala R., Krzyżanowski J., Matuszek J.: Technologiczne przygotowanie produkcji. Materiały konferencji „Automatyzacja produkcji 2000. Wiedza-technika-postęp". Prace Naukowe Instytutu Technologii Maszyn i Automatyzacji Politechniki Wrocławskiej nr 78. Konferencje. Referaty plenarne. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2000, s. 45-78.
27. Chlebus E., Dybała B., Boratyński T.: Geometric CAD models in design and integrated product development. Proc. of the CAMT Conference "Modern trends in manufacturing", Wrocław 2002. pp. 273-282.
28. Chlebus E.: Techniki komputerowe CAx w inżynierii produkcji. WNT, Warszawa 2000.
29. Choset H.: Coverage of Known Spaces: The Boustrophedon Cellular Decomposition. Autonomous Robots, 9 (2000), pp. 247-253. Kluwer Academic Publishers 2000.
30. Cichosz P.: Systemy uczące się. WNT, Warszawa 2000.
31. Coiffet P.: Interaction with the environment: robot sensors and sensing. Kogan Page, London 1987.
32. COSIMIR Professional: www.cosimir.com
33. Crowley J.L., Causse O., Reignier P.: Layers of Control in Autonomous Navigation. in Mechatronics and Robotics, I. Advances in Design and Manufacturing, Edts: MacConaill P.A., Drews P., Robrock K.-H., IOS Press, Amsterdam 1991, pp. 67-76.
34. Dalventhal K.M., Kissner A., Kulick M.: Matematyka. Kompendium. Horyzont, Grupa Wydawnicza Bertelsmann Media 2002.
35. Daman D., Kari S., Bade A., Suaib N.M.: An Overview of Collision Detection in Virtual Environment. Proc. of The 18th International Conference on CAD/CAM, Robotics and Factories of The Future. Porto, vol. 2, Portugal 2002, pp. 501-513.

36. Dapena E.G., Moreno L.L., Nombela A.P.: path Planning with Uncertainty. Proc. of The 18th International Conference on CAD/CAM, Robotics and Factories of The Future. Porto, vol. 2, Portugal 2002, pp. 491-499.
37. Deok-Soo K., Donguk K., Sugihara K.: Voronoi diagram of a circle set from Voronoi diagram of a point set: I. Topology. Computer Aided Geometric Design, 18 (2001), pp. 541-562.
38. Diéguez A.R., Sanz R., López J.: Deliberative On-Line Local Path Planning for Autonomous Mobile Robots. Journal of Intelligent and Robotics Systems, 37 (2003), pp. 1-19. Kluwer Academic Publishers 2003.
39. Dillmann R., Schneider S.: Integration of Product Data Models Into Object-Oriented Robot Assembly Planning and Programming Systems. in echatronics and Robotics, I. Advances in Design and Manufacturing, Edts: MacConaill P.A., Drews P., Robrock K.-H., IOS Press, Amsterdam 1991, pp. 127-141.
40. Duelen G., Münch H., Zhang Y.: Planning Systems for Robot Motion Optimization. Prod. Engineering Research and Development in Germany. Annals of the German Academic Society for Production Engineering, vol 1, Issue 2, 1994. Hanser Verlag 1994.
41. Dulęba I., Łysakowska B.: Zmodyfikowana metoda R.H. Taylora śledzenia ścieżek prostoliniowych. t. 2, Materiały III Krajowej Konferencji Robotyki, Wrocław 1990, s. 61-67.
42. Dulęba I.: Algorithms of Motion Planning for Nonholonomic Robots. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 1998.
43. Dulęba I.: Metody i algorytmy planowania ruchu robotów mobilnych i manipulacyjnych. Akademicka Oficyna Wydawnicza EXIT, Warszawa 2001.
44. Durrant-Whyte H.F.: Integration, coordination and control of multi-sensor robot system. Kluwer Academic, Boston 1988.
45. Egerstedt M., Martin C.F.: Optimal trajectory planning and smooth splines. Automatica 37(2001), Pergamon Press 2002, pp. 1057-1064.
46. Elnagar A., Hussein A.: On optimal constrained trajectory planning in 3D environments. Robotics and Autonomous Systems. Elsevier, 33(2000), pp. 195-206.
47. eMPowers (Robocad): www.tecnomatix.com/showpage.asp

48. Etzion M., Rappoport A.: Computing Voronoi skeleton of a 3-D polyhedron by space subdivision. Computational Geometry. Theory and Applications, 21 (2002), pp. 87-120.
49. Ferrari C., Pagello E., Ota J., Arai T.: Multirobot motion coordination in space and time. Robotics and Autonomous Systems 25(1998), Elsevier 1998, pp. 219-229.
50. Flash T., Richardson M.J.E., Handzel A.A.: The Control of Motion: From Biological to Robotic Systems. Proc. of the Sixfth International Symposium on Methods and Models in Automation and Robotics. Międzyzdroje 2000, s. 523-531.
51. Forsythe W., Goodall R.M.: Digital Control. McGraw-Hill, Inc.
52. Fortuna Z., Macukow B., Wąsowski J.: Metody numeryczne. Wyd. II, WNT, Warszawa 1993.
53. Frey W.: RAPID – Język programowania robotów przemysłowych. Narzędzia wspierające programowanie. Materiały VI Krajowej Konferencji Robotyki. Świeradów Zdrój 1998, s. 243-249.
54. Gajek L., Kałuszka M.: Wnioskowanie statystyczne. Modele i metody. WNT, Warszawa 1996.
55. Galicki M.: Generowanie bezkolizyjnych trajektorii manipulatorów redundantnych. Studia z Automatyki i Informatyki, t. 26, Poznańskie Towarzystwo Przyjaciół Nauk. Wydział Nauk Technicznych. Prace Komisji Informatyki i Automatyki, Poznań 2001, s. 55-68.
56. Galicki M.: Wybrane metody planowania optymalnych trajektorii robotów manipulacyjnych. WNT, Warszawa 2000.
57. Gavrilova M.L.: On a Nearest-Neighbor Problem Under Minkowski and Power Metric for Large Data Sets. The Journal of Supercomputing, 22 (2002), pp. 87-98. Kluwer Academic Publishers 2002.
58. Gawłowicz P.: Zastosowanie metod przeszukiwania heurystycznego do wyznaczania optymalnego sterowania robotów. Studia z Automatyki i Informatyki, t. 23, Poznańskie Towarzystwo Przyjaciół Nauk. Wydział Nauk Technicznych. Prace Komisji Informatyki i Automatyki, Poznań 1998, s. 113-124.
59. Giacomo De G., Lespérance Y., Levesque H.J.: *ConGolog*, a concurrent programming language based on the situation calculus. Artificial Intelligence 121(2000), Elsevier 2000, pp. 109-169.

60. Giergiel M., Hendzel Z., Żylski W.: Modelowanie i sterowanie mobilnych robotów kołowych. Wydawnictwo Naukowe PWN, Warszawa 2002.
61. Glasius R., Komoda A., Gielen S.C.A.: Neural Network Dynamics for Path Planning and Obstacle Avoidance. Neural Networks, Pergamon Press, Vol 8, No. 1, pp. 125-133, 1995.
62. Hailu G.: Symbolic structures in numeric reinforcement for learning optimum robot trajectory. Robotics and Autonomous Systems 37(2001), Elsevier 2001, pp. 53-68.
63. Hemmerle J.S., Prinz F.B.: Next Generation Environment for Manufacturing Task Planning. in Mechatronics and Robotics, I. Advances in Design and Manufacturing, Edts: MacConaill P.A., Drews P., Robrock K.-H., IOS Press, Amsterdam 1991, pp.91-101.
64. Honczarenko J.: Elastyczna automatyzacja wytwarzania. Obrabiarki i systemy obróbkowe. WNT, Warszawa 2000.
65. Honczarenko J.: Roboty przemysłowe. Budowa i zastosowanie. WNT, Warszawa 2004.
66. Honczarenko J.: Roboty przemysłowe. Elementy i zastosowanie. WNT, Warszawa 1996.
67. IGRIP: www.aatinc.on.ca/robots.htm
68. Iosifescu M.: Skończone procesy Markowa i ich zastosowanie. PWN, Warszawa 1988.
69. Jacak W.: Roboty inteligentne. Metody planowania działań i ruchu. Wydawnictwo Politechniki Wrocławskiej, Wrocław 1991.
70. Jamhour E., André P.J.: Planning smooth trajectories along parametric path. Mathematics and Computers in Simulation. Elsevier N.H., 41(1996), pp. 615-626.
71. Jankowski B.: GRAFY. Algorytmy w Pascalu. Wydawnictwo MIKOM, Warszawa 1998.
72. Jankowski B.: Programowanie w praktyce. Wydawnictwo MIKOM, Warszawa 1999.
73. Kandil A.A., Nour Eldin H.A., Markova V.: A. Quaternionic Formulation For The Path Geometry of A redundant Planar Robot With Synchronized Angle Motion. Proc. of the Sixfth International Symposium on Methods and Models in Automation and Robotics. Międzyzdroje 2000, s. 703-709.
74. Kandil A.A., Nour Eldin H.A.: Types, Properties and Parameterization of The Possible Planar Path Generated by Synchronized Angle Motion of a 2 DOF Planar Robot. Proc. of the Sixfth International Symposium on Methods and Models in Automation and Robotics. Międzyzdroje 2000, s. 697-671.
75. Katevas N.I., Tzafestas S.G., Pnevmatikatos Ch.G.: The Approximate Cell Decomposition with Local Node Refinement Global Path Planning Method: Path Nod Refinement and Curve Parametric Interpolation. Journal of Intelligent and Robotics Systems, 22 (1998), pp. 289-314. Kluwer Academic Publishers 1998.
76. Kavraki L.E., Latombe J.-C., Motwani R., Raghavan P.: Randomized Query Processing in Robot Path Planning. Journal of Computer and System Sciences, 57(1998), pp. 50-60.
77. Khoo L.P., Teo M.Y.: A prototype fuzzy-based classifier system for trajectory planning. Journal of Network and Computer Applications 20(1997), Academic Ptess Ltd. 1997, pp. 191-202.
78. Kingston J.: High Performance Knowledge Bases: four approaches to knowledge acquisition, representation and reasoning for workaround planning. Expert Systems with Applications, Pergamon Press, 21(2001), pp. 181-190.
79. Knapczyk J., Lebiediew P.A.: Teoria mechanizmów przestrzennych i manipulatorów. WNT, Warszawa 1990.
80. Koch T.: Programowanie redundantnych robotów przemysłowych. Prace Naukowe Instytutu Technologii Maszyn i Automatyzacji Politechniki Wrocławskiej, Seria: Monografie, z. 18, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 1996.
81. Kohut P., Uhl. T.: The rapid prototyping of the visual servoing on MATLAB/Simulink/dSPACE environment. Proceedings of the 7th IEEE International Conference on Methods and Models in Automation and Robotics. Vol. 2, Wydawnictwo Uczelniane Politechniki Szczecińskiej, Międzyzdroje 2001, pp. 677-682.
82. Kohut P., Uhl. T.: Szybkie prototypowanie układów sterowania wizyjnego w środowisku MATLAB/Simulink/dSPACE. Materiały VII Krajowej Konferencji Robotyki. Oficyna Wydawnicza Politechniki Wrocławskiej, Lądek Zdrój 2001, s. 117-126.
83. Kowalczuk Z., Ostachowicz W., Raczyński P., Roznowski G.: Badania modelowe robota podwodnego. Materiały Seminarium Naukowego. „Projekty badawcze w dziedzinie robotyki". PIAP, Warszawa 1994, s. 205-210.

84. Kosmol J.: Automatyzacja obrabiarek i obróbki skrawaniem. WNT, Warszawa 1995.
85. Kost G.: Collision-Free Robot's Trajectory Planning Based Upon Self-Learning Algorithm Assumption. in Computer Integrated Manufacturing. Advanced Design and Management, Edts: B.Skołud, D. Krenczyk. WNT, Warszawa 2003, pp.287-291.
86. Kost G.: Układy sterowania robotów przemysłowych. Wyd. Politechniki Śląskiej, Gliwice 2000.
87. Kost G.G., Skołud B.: Definiowanie struktur kinematycznych. Materiały VI Krajowej Konferencji Robotyki Świeradów Zdrój, Oficyna Wydawnicza Politechniki Wrocławskiej, seria Prace Naukowe ICT, Wrocław 1998, 9-12.09.1998. t. 1, s. 27-34.
88. Kost G.G., Stankiewicz K.: Stochastic Markov's decision making process In the analysis of robot working space. Proc. of the Symposium on Methods of Artificial Intelligence AI-METH-2003, Edts.: T. Burczyński, W. Cholewa, W. Moczulski, Gliwice 2003, pp.70-72.
89. Kost G.G., Zdanowicz R.: Monitoring and Simulation in The Model of The Computer Turning Cell, Materiały II Międzynarodowej Konferencji CIM'94, Zakopane 1994, Zeszyty Naukowe Politechniki Śląskiej z. 117, Gliwice 1994, s. 403-414.
90. Kost G.G.: Graf obsługi zrobotyzowanego gniazda wytwórczego. Materiały VIII Międzynarodowej Konferencji Naukowo-Technicznej pt.: Tendencje rozwojowe w procesach produkcyjnych. Zielona Góra 1997, s. 175-180.
91. Kost G.G.: Język opisu sceny robota w systemie komputerowego wspomagania programowania robotów przemysłowych. IV Krajowa Konferencja Robotyki, Wrocław 1993, t. 1, s. 343-351.
92. Kost G.G.: Method of avoiding collisions of a robot with its environment based on the system of weights. Proc. of the 7th International Conference ROBTEP 2004-Automation/Robotics in theory and practice, Vyšné Ružbachy 2004, pp. 301-304, Slovak Republic.
93. Kost G.G.: Metoda unikania przeszkód w systemie automatycznego programowania robotów przemysłowych. Materiały Konferencji Naukowo–Technicznej Automatyzacja – nowości – perspektywy AUTOMATION 2004, PIAP, Warszawa 2004, s. 543-551.
94. Kost G.G.: Model zrobotyzowanego elastycznego systemu wytwórczego. Materiały III Konferencji Naukowo-Technicznej MECHATRONIKA '97. Warszawa 1997, s. 115-120.

95. Kost G.G.: Modelowanie stanu robota z zastosowaniem sieci Petriego. Materiały V Krajowej Konferencji Robotyki, Wrocław - Świeradów Zdrój 1996, s. 172-180.
96. Kost G.G.: Problem Oriented Programming of the Robot. Model of the Technological Process. Proc. of the CIM'96 Conference, Zakopane 1996, pp. 183-192.
97. Kost G.G.: Strategia poszukiwanie bezkolizyjnych ścieżek robotów. Materiały 8 Krajowej Konferencji Robotyki, Polanica Zdrój 2004, w druku. (Preprinty: s. 229-230).
98. Kost G.G.: Strategy of determining non-collision trajectories of robot operating in a complex process environment. Proc. of the 7th International Conference ROBTEP 2004-Automation/Robotics in theory and practice, Vyšné Ružbachy 2004, pp. 305-308, Slovak Republic.
99. Kost G.G.: System of designing robot trajectory on the grounds of Markov's decision processes and Q-learning algorithm. Proc. of the 7th International Conference ROBTEP 2004-Automation/Robotics in theory and practice, Vyšné Ružbachy 2004, pp. 309-312, Slovak Republic.
100. Kost G.G.: Zastosowanie algorytmu Q-learning do programowania robotów przemysłowych. Proc.of the 12th International Scientific Conference on Achievements in Mechanical and Materials Engineering AMME'2003. Edt. L.A. Dobrzański, Gliwice – Zakopane 2003, pp. 505-508.
101. Kowalenko I.N., Kuzniecow N.J., Szurienkow W.M.: Procesy stochastyczne. Poradnik. PWN, Warszawa 1989.
102. Kozłowski K., Dutkiewicz P., Wróblewski W.: Modelowanie i sterowanie robotów. Wydawnictwo Naukowe PWN, Warszawa 2003.
103. Kreczmer B.: Multi-Sonar System for Mobile Robot Navigation. Proc. of the Fifth International Symposium on Methods and Models in Automation and Robotics. Międzyzdroje 1998, s. 931-936.
104. Krishnan R., Das A., Gurumoorthy B.: Octree encoding of B-RepBased Objects. Computer and Graphics, Vol.20, No. 1, pp.107-114, Pergamon 1996.
105. Kröse B.J.A.: Learning from delayed rewards. Robotics and Autonomous Systems 15(1995), Elsevier 1995, pp. 233-235.
106. Kruusmaa M.: Global Level Path Planning for Mobile Robots in Dynamic Environments. Journal of Intelligent and Robotics Systems, 38 (2003), pp. 55-83. Kluwer Academic Publishers 2003.

107. Kulikowski J. L.: Zarys teorii grafów. Zastosowania w technice. Biblioteka Naukowa Inżyniera. PWN, Warszawa 1986.
108. Kushner H.: Wprowadzenie do teorii sterowania stochastycznego. PWN, Warszawa 1983.
109. Kwolek B., Wysocki M.: Trajectory Planning and Stabilizing Using Visual Feedback and Transputer Network. Proc. of the Fourth International Symposium on Methods and Models in Automation and Robotics. Międzyzdroje 1997, s. 921-926
110. Latombe J.C.: Robot Motion Planning. Kluwer Academic Publisher, 1993.
111. Laumond J.P. (Ed): Robot Motion Planning and Control. Springer-Verlag, 1998.
112. Lee D.T., Yang C.D., Wong C.K.: Rectilinear paths among rectilinear obstacles. Discrete Applied Mathematics 70(1996), Elsevier N.H., pp. 185-215.
113. Lee M.H.: Intelligent Robotics. Halsted Press (John Wiley&Sons)& Open University Pres (Milton Keynes), New York 1989.
114. Leng D.Y., Chen M.: Robot Trajectory Planning Using Simulation. Robotics and Computer-Integrated Manufacturing, Pergamon 1997, Vol.13, No. 2, pp.121-127.
115. Lin F., Levesque H.J.: What robots can do: robot programs and effective achievability. Artificial Intelligence 101(1998), Elsevier 1998, pp. 201-226.
116. Littman M.L., Sutton R.S., Singh S.: Predictive Representation od State. Proceedings of the Conference on Advances in Neural Information Processing Systems 14 (2001), pp.7, http://www-anw.cs.umass.edu/~rich/publication.html
117. Lowrie T.: The influence of visual and spatial reasoning in interpreting simulated 3-D works. International Journal of Computers for Mathematical Learning. 7 (2002), pp. 301-318. Kluwer Academic Publishers 2002.
118. Lynch M.: Computer Numerical Control. Advanced Techniques. McGra-Hill, Inc. 1993.
119. Maaref H., Barret C.: Sensor-based navigation of mobile robot in an indoor environment. Robotics and Autonomous Systems 38(2002), Elsevier 2002, pp. 1-18.
120. Majchrzak E., Mochnacki B.: Metody numeryczne. Podstawy teoretyczne, aspekty praktyczne i algorytmy. Wyd. Politechniki Śląskiej, Gliwice 1996.
121. Mansor M.A., Morris A.S.: Path Planning in Unknown Environment With Obstacles Using Virtual Window. Journal of Intelligent and Robotics Systems, 24 (1999), pp. 235-251. Kluwer Academic Publishers 1999.

122. Marsland S., Shapiro J., Nehmzow U.: A self-organising network that grows when required. Neural Networks, 15 (2002), pp. 1041-1058. Pergamon 2002.
123. Marti K., Qu S.: Path Planning for Robots by Stochastic Optimization Methods. Journal of Intelligent and Robotics Systems, 22 (1998), pp. 117-127. Kluwer Academic Publishers 1998.
124. Martin C.F., Sun Sh., Egerstedt M.: Optimal Control, Statistics and Path Planning. Mathematical and Computer Modelling. 33(2001), Pergamon Press 2002, pp. 23-253.
125. Martin P., Millán J. del R.: Robot arm reaching through neural inversions and reinforcement learning. Robotics and Autonomous Systems 31(2000), Elsevier 2000, pp. 227-246.
126. Martinez-Alfaro H., Gómez-Garcia S.: Mobile robot path planning and tracking using simulated annealing and fuzzy logic control. Expert Systems with Applications 15(1998), Pergamon Press 1998, pp. 421-429.
127. Mediavilla M., Fraile J-C., Galindo I.J., Gonzáles M.T.: Selection of Strategies for Collision-free Motion in Multi-manipulator Systems. Journal of Intelligent and Robotics Systems, 38 (2003), pp.85-104. Kluwer Academic Publishers 2003.
128. Millán J. del R.: Reinforcement learning of goal-directed obstacle-avoiding reaction strategies in an autonomous mobile robot. Robotics and Autonomous Systems 15 (1995), pp. 275-299.
129. Mirolo C., Pagello E.: A New Cell-subdivision Approach to Plan Free Translations in Cluttered Environments. Journal of Intelligent and Robotics Systems, 38 (2003), pp. 5-30. Kluwer Academic Publishers 2003.
130. Miyazaki K., Yamamura M., Kobayashi Sh.: *k*-Certainty Exploration Method: an action selector to identify the environment in reinforcement learning. Artificial Intelligence 91(1997), Elsevier 1997, pp. 155-171.
131. Morecki A., Knapczyk J., Kędzior K.: Teoria mechanizmów i manipulatorów. Podstawy i przykłady zastosowań. WNT, Warszawa 2002.
132. Morecki A., Knapczyk J.: Podstawy robotyki. Teoria i elementy manipulatorów i robotów. WNT, Warszawa 1993.
133. Nagabhushan P., Manohara Pai M.M.: Cognition of free space for planning the shortest path: A framed free space approach. Pattern Recognition Letters 22(2001), Elsevier N.H. 2002, pp. 971-982.

134. Ngoc-Minh L.: Abstract Voronoi diagram in 3-space. Journal of Computer and System Sciences. 68 (2004), pp. 41-79. Elsevier 2004.
135. Nof Sh.: Handbook of Industrial Robotics. John Wiley& Sons, Inc. 2000.
136. Ohsawa Y., Tamura K.: Efficient Location for a Semi-Obnoxious Facility. Annals of Operations Research, 123 (2003), pp. 173-188. Kluwer Academic Publishers 2003.
137. Olszewski M.: Manipulatory i roboty przemysłowe. WNT, Warszawa 1992.
138. Ostachowicz W., Szwedowicz J.: Analiza dynamiki manipulatora z uwzględnieniem podatności jego członów. Materiały II Krajowej Konferencji Robotyki, Wrocław 1988, s. 179-186.
139. Ostachowicz W., Szwedowicz J.: Dynamika manipulatorów z uwzględnieniem podatności i luzów w układach napędowych. Materiały XXVII Sympozjum „Modelowanie w mechanice". Wyd. Politechniki Śląskiej, Gliwice 1988, s.377-384.
140. Pająk G., Galicki M.: Collision-Free Trajectory Planning of The Redundant Manipulators. Proc. of the Sixth International Symposium on Methods and Models in Automation and Robotics. Międzyzdroje 2000, s. 605-610.
141. Pająk G.: Planowanie bezkolizyjnej trajektorii manipulatora przy ograniczeniach na sterowania. Studia z Automatyki i Informatyki, t. 26, Poznańskie Towarzystwo Przyjaciół Nauk. Wydział Nauk Technicznych. Prace Komisji Informatyki i Automatyki, Poznań 2001, s. 147-158.
142. Pająk I.: Planowanie suboptymalnych trajektorii manipulatorów redundantnych w przestrzeni roboczej z przeszkodami. Studia z Automatyki i Informatyki, t. 26, Poznańskie Towarzystwo Przyjaciół Nauk. Wydział Nauk Technicznych. Prace Komisji Informatyki i Automatyki, Poznań 2001, s. 135-145.
143. Pawlukowicz P., Jardzioch A., Honczarenko J.: Industrial Robot Programming in Production System Using off-line Method. Proc. of the Sixth International Symposium on Methods and Models in Automation and Robotics. Międzyzdroje 2000, s. 661-664.
144. Petiot J.F., Chedmail P., Hascoët J.-Y.: Contribution to the scheduling of trajectories in robotics. Robotics and Computer-Integrated Manufacturing 14(1998), Pergamon Press 1998, pp. 237-251.
145. Pfeifer R.: Cognition – Perspectives from autonomous agents. Robotics and Autonomous Systems 15(1995), Elsevier 1995, pp. 47-70.
146. Pieniążek A., Weiss J., Winiarz A.: Procesy stochastyczne w problemach i zadaniach. Wyd. Politechniki Krakowskiej, Kraków 2000.

147. Plucińska A., Pluciński E.: Probabilistyka. Rachunek prawdopodobieństwa. Statystyka matematyczna. Procesy stochastyczne. WNT, Warszawa 2000.
148. Podgórska M., Śliwka P., Topolewski M., Wrzosek M.: Łańcuchy Markowa w teorii i w zastosowaniach. Szkoła Główna Handlowa. Oficyna Wydawnicza, Warszawa 2002.
149. Podsędkowski L.: Dynamiczne planowanie trajektorii robotów mobilnych w zmiennej przestrzeni roboczej. Zeszyty Naukowe Politechniki Łódzkiej nr 809, Wyd. Politechniki Łódzkiej, Łódź 1999.
150. Podsiadło M.: Zastosowania algorytmów genetycznych do generacji trajektorii robotów. Studia z Automatyki i Informatyki. t. 24, Poznańskie Towarzystwo Przyjaciół Nauk. Wydział Nauk Technicznych. Prace Komisji Informatyki i Automatyki, Poznań 1999, s. 137-150.
151. Preparata F.P., Shamos M. I.: Geometria obliczeniowa. Wprowadzenie. Wyd. Helion, Gliwice 2002.
152. Pritschow G.: Technika sterowania obrabiarkami i robotami przemysłowymi. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 1995.
153. Pugh A.: Robot sensors. Vol. 1, Vision. IFS, Kempston 1986.
154. Pugh A.: Robot sensors. Vol. 2, Tactile and non-vision. IFS, Kempston 1986.
155. Putz B.: Znaczenie punktów kontrolnych w procesie projektowania powierzchni. Mechanik nr 12/2003, s. 747-750.
156. Rafanidis I., Vlahavas I.: Multiobjective heuristic state-space planning. Artificial Intelligence 145(2003), Elsevier 2003, pp. 1-32.
157. Rembold U.: Robot Technology and Applications. Marcel Dekker, Inc. 1990.
158. Rieger J.H.: Voronoi Diagrams of Real Algebraic Sets. Geometriae Dedicata, 98 (2003), pp. 81-94. Kluwer Academic Publishers 2003.
159. Robotmaster: www.robotmaster.com
160. Robsim – CAMELOT: www.camelot.dk
161. Roszkowska E., Wnuk M.: „CeRo" – Implementacja języka programowania robotów w rozproszonym środowisku obliczeniowym. Materiały VI Krajowej Konferencji Robotyki, Świeradów Zdrój 1998, s. 267-278.
162. Samejima K., Omori T.: Adaptive internal state space construction method for reinforcement learning of a real-world agent. Neural Networks 12(1999), Pergamon Press 1999, pp. 1143-1155.

163. Saona-Vázquez C., Navazo I, Brunet P.: The visibility octree: a data structure for 3-D navigation. Computers & Graphics, 23 (1999), pp. 635-643. Pergamon 1999.
164. Saramago S.F.P., Junior St. V.: Optimal trajectory planning of robot manipulators in the presence of moving obstacles. Mechanism and Machine Theory. Pergamon Press, 35 (2000), pp. 1079-1094.
165. Saramago S.F.P., Junior St. V.: Optimization of The Trajectory Planning of Robot Manipulators Taking Into Account The Dynamic of The System. Mechanism and Machine Theory. Vol. 33, No. 7, pp. 883-894. Pergamon Press 1988.
166. Sawik T.: Optymalizacja dyskretna w elastycznych systemach produkcyjnych. WNT, Warszawa 1992.
167. Sawik T.: Planowanie i sterowanie produkcji w elastycznych systemach produkcyjnych. WNT, Warszawa 1996.
168. Shah Hamzei G.H., Mulvaney D.J.: On-line learning of fuzzy decision trees for global path planning. Engineering Applications of Artificial Intelligence 12(1999), Pergamon Press 1999, pp. 93-109.
169. Shehory O., Kraus S.: Methods for task allocation via agent coalition formation. Artificial Intelligence 101(1998), Elsevier 1998, pp. 165-200.
170. Shiller Z., Sundar S.: On-Line Dynamic Motion Planning. Proc. of the Fourth International Symposium on Methods and Models in Automation and Robotics. Międzyzdroje 1997, s. 897-906.
171. Sierocki I.: Jak układać klocki: Wprowadzenie do teorii reprezentacji problemu planowania akcji. Materiały IV Krajowej Konferencji Robotyki. t.1, Wydawnictwo Politechniki Wrocławskiej, Wrocław 1993, s. 273-280.
172. Skołud B., Kost G. G.: GUI - system for computer assisted programming of industrial robots. Elektrotechnik und Informationstechnik ÖVE - Verbandszeitschrift, No 5, 1998, pp. 251-256.
173. Skołud B., Kost G.: Planowanie bezkolizyjnych trajektorii robota metodą modelowania oktalnego. Materiały IV Krajowej Konferencji Robotyki. t.1, Wydawnictwo Politechniki Wrocławskiej, Wrocław 1993, s. 281-289.
174. Skołud B., Kost G.G.: Computer Aided Detection of Solids Collision in FMS With Robot by CAD Simulation. Materiały Międzynarodowej Konferencji CIM'92, Zakopane 1992. Zeszyty Naukowe Politechniki Śląskiej, Gliwice 1992, s. 359-366.

175. Skołud B., Kost G.G.: Computer Model of FMC. International DAAAM Symposium "Automation and Metrology: Challange and Chance", Maribor Slovenia 1994.
176. Skołud B., Kost G.G.: Information flow in the flexible manufacturing cell (FMC). Proc. of the III-rd International Scientific Conference AMME'94, Gliwice 1994, s.311-316.
177. Spong M.W., Vidyasagar M.: Dynamika i sterowanie robotów. WNT, Warszawa 1997.
178. Steels L.: When are robots intelligent autonomous agents? Robotics and Autonomous Systems 15(1995), Elsevier 1995, pp. 3-9.
179. Stone P., Sutton R.S.: Scalling Reinforcement Learning Toward RoboCup Soccer. Proceedings of the 18th International Conference on Machine Learning, 2001; http://www-anw.cs.umass.edu/~rich/publication.html
180. Sutton R.S., Barto A.G.: Reinforcement Learning: An Introduction. MIT Press 1998; http://www-anw.cs.umass.edu/~rich/book
181. Sutton R.S.: Reinforcement Learning and Artificial Intelligence. pp. 19, http://www-anw.cs.umass.edu/~rich/publication.html
182. Sutton R.S.: Reinforcement Learning Architectures. Proceedings ISKIT'92 International Symposium on Neural Information Processing. Fukuoka, Japan 1992; http://www-anw.cs.umass.edu/~rich/publication.html
183. Świder J. (red.): Sterowanie i automatyzacja procesów technologicznych i układów mechatronicznych. Układy pneumatyczne i elektropneumatyczne ze sterowaniem logicznym (PLC). Wydawnictwo Politechniki Śląskiej, Gliwice 2002.
184. Świder J., Wojnarowski J., Kropka J.: Estimation of Industrial Robots Subsystems Vibrations Using Transformation of Hybrid Graphs into Bond Graphs. Tenth World Congress on the Theory of Machines and Mechanisms. University of Oulu, Finland, June 20-24,1999.
185. Świder J., Wojnarowski J.: Estimation of the Accuracy of Setting an Industrial Robot with Regard to External Influence, Ninth World Congress on the Theory of Machines and Mechanisms, Politecnico di Milano, Italy, August 29/September 2, 1995, IFToMM, Proceedings vol.3, p.2117÷2121.

186. Świder J., Wojnarowski J.: Numerical Computation of the Accuracy of a Trajectory of a Three - Dimensional Model of an Industrial Robot by Means of Matrix Hybrid Graphs. Proceedings of the Second ECPD International Conference on Advanced Robotics, Intelligent Automation and Active Systems. Vienna, Austria, September 26 - 28, 1996, p. 486-491.
187. Świder J.: Generowanie równań ruchu metodą grafów. Konferencja Mechaniki Analitycznej, 26-27 września 1995, Instytut Techniki Lotniczej i Mechaniki Stosowanej, Warszawa. (Streszczenia).
188. Świder J.: Macierzowe grafy hybrydowe w opisie drgających, złożonych układów mechanicznych, ZN Politechniki Śląskiej, Mechanika, z.106, Gliwice 1991, s.221.
189. Tadeusiewicz R.: Systemy wizyjne robotów przemysłowych. WNT, Warszawa 1992.
190. Tchoń K., Mazur A., Dulęba I., Hossa R., Muszyński R.: Manipulatory i roboty mobilne. Modele, planowanie ruchu, sterowanie. Akademicka Oficyna Wydawnicza PLJ, Warszawa 2000.
191. Thrun S., Mitchell T.M.: Lifelong robot learning. Robotics and Autonomous Systems 15(1995), Elsevier 1995, pp. 25-46.
192. Thrun S.: An approach to learning mobile robot navigation. Robotics and Autonomous Systems, vol. 15 (1995), pp. 301-319. Elsevier 1995.
193. Ting Y., Lei W.I., Jar H.C.: A path planning alghoritm for industrial robots. Computers and Industrial Engineering 42(2002), Pergamon Press 2002, pp. 299-308.
194. Tönshoff H.K., Inasaki I.: Sensors in manufacturing. Wiley-VCH, Weinheim 2001.
195. Torras C.: Robot adaptivity. Robotics and Autonomous Systems 15(1995), Elsevier 1995, pp.11-23.
196. Tsitsiklis J.N.: On the Convergence of Optimistic Policy Iteration. Journal of Machine Learning Research 3(2002), pp. 59-72.
197. Tsoularis A., Kambhampati C.: On-line Planning for Collision Avoidance on the Nominal Path. Journal of Intelligent and Robotic Systems, 21 (1998), pp. 327-371. Kluwer Academic Publishers 1998.
198. Ueda K.: Reinforcement Learning Approaches to Biological Manufacturing Systems. Annals of the CIRP Vol. 49/1/2000.
199. Uhl T.: Komputerowe wspomaganie projektowania mechatronicznego. Materiały Konferencji Naukowo–Technicznej „Automatyzacja – nowości – perspektywy" AUTOMATION 2002", PIAP, Warszawa 2002, s. 41-52.

200. Uhl T.: Komputerowo wspomagana identyfikacja modeli konstrukcji mechanicznych. WNT, Warszawa 1997.
201. Uhl. T., Giergiel J.: Identyfikacja układów mechanicznych. PWN, Warszawa 1990.
202. Uhl.T.:, Bojko T., Mrozek Z., Petko M., Szwabowski W.: Mechatroniczne podejście do projektowania i implementacji układów sterowania. Materiały I Konferencji „Automatyzacja maszyn, urządzeń i procesów" APRO'99. Krynica 1999, Gutenberg - Kraków, s. 239-250.
203. Uptal R., Yaoxian X.: 3-D object decomposition with extended octree model and its application in geometric simulation of NC machining. Robotics and Computer-Integrated Manufacturing, 14 (1998), pp. 317-327, Pergamon 1998.
204. Vörös J.: A strategy for repetitive neighbor finding in octree representations. Image and Vision Computing, 18 (2000), pp. 1085-1091, Elsevier 2000.
205. Vörös J.: Low-cost implementation of distance maps for path planning using matrix quadtrees and octrees. Robotics and Computer-Integrated Manufacturing, 17(2001), Pergamon Press 2001, pp. 447-459.
206. Warczyński J.: Programowanie niezawodnego ruchu manipulacyjnego robota w warunkach niepewności. Studia z Automatyki i Informatyki, t.22, Poznańskie Towarzystwo Przyjaciół Nauk. Wydział Nauk Technicznych. Prace Komisji Informatyki i Automatyki, Poznań 1997, s. 101-115.
207. Warczyński J.: Sterowanie ruchem adaptacyjnym robota. Studia z Automatyki i Informatyki, t. 23, Poznańskie Towarzystwo Przyjaciół Nauk. Wydział Nauk Technicznych. Prace Komisji Informatyki i Automatyki, Poznań 1998, s. 179-189.
208. Weck M., Dammertz R.: OPERA – A New Approach to Robot Programming. Annals of the CIRP, Vol. 44/1/1995, pp. 389-392.
209. Weck M., Weeks J., Peper St.: Programming of Industrial Robots. Production Engineering, Vol. 1, No. 2, 1994, pp. 113-116.
210. Weiβ G.: Distributed reinforcement learning. Robotics and Autonomous Systems 15(1995), Elsevier 1995, pp. 135-142.
211. Whitehead S.D., Lin L-J.: Reinforcement learning of non-Markov decision processes. Artificial Intelligence 73(1995), Elsevier 1995, pp. 271-306.
212. www.directindustry.com/find/robot_off_line_programming_software.html
213. Yahja A., Singh S., Stentz A.: An efficient on-line path planner for outdoor mobile robots. Robotics and Autonomous Systems 32(2000), Elsevier 2000, pp. 129-143.

214. Yang S.X., Meng M.: An efficient neural network approach to dynamic robot motion planning. Neural Networks 13(2000), Pergamon Press 2000, pp. 143-148.
215. Young K.-Y., Huang Ch.-Y.: Collision-Free Path Planning and Modification Based on Task Requirements. Mechanism and Machine Theory. Pergamon Press 1997, Vol. 32, No. 1, pp. 21-37.
216. Zanardi Ch., Herve J.-Y., Cohen P.: Representation of interactions between mobile robots for trajectory planning. Control Engineering Practice 7(1999), Pergamon Press 1999, pp. 753-761.
217. Zha X.F.: Optimal pose trajectory planning for robot manipulators. Mechanism and Machine Theory 37(2002, Pergamon Press 2002, pp. 1063-1086.
218. Zhang W.J., Liu S.N., Li Q.: Data/knowledge representation of modular robot and its working environment. Robotics and Computer Integrated Manufacturing. 16(2000), Pergamon Press 2000, pp. 143-159.
219. Zhou Ch.: Robot learning with GA-based fuzzy reinforcement learning agents. Information Sciences 145(2002), Elsevier N.H. 2002, pp. 45-68.
220. Zhu Z.H., Mayorga R.V., Wong A.K.C.: Dynamic Robot Manipulator Trajectory Planning for Obstacle Avoidance. Mechanics Research Communications. Vol. 26, No. 2, pp. 139-144. Pergamon Press 1999.
221. Zwiernik P. W.: Wstęp do ukrytych modeli Markowa i metody Bauma-Welcha, s. 29; http://akson.sghwaw.pl/~pzwier/sci/hmm.pdf

PLANOWANIE BEZKOLIZYJNYCH ŚCIEŻEK MANIPULACYJNYCH I STACJONARNYCH ROBOTÓW PRZEMYSŁOWYCH OPARTE NA PROCESACH DECYZYJNYCH MARKOWA I FUNKCJI OCEN

Streszczenie

Przedstawiona rozprawa jest wynikiem kilkuletnich badań nad problematyką planowania bezkolizyjnych trajektorii manipulacyjnych robotów przemysłowych. Stanowi próbę rozwiązania tego zagadnienia w obszarze planowania bezkolizyjnych ścieżek robotów w oparciu o powszechne w robotyce stacjonarnej metody graficzne. W pracy zaprezentowano nowatorskie rozwiązanie oparte na adaptacji wysoko wydajnych obliczeniowo i czasowo technik planowania ruchu robotów mobilnych do zadania planowania ruchu przemysłowych robotów manipulacyjnych i stacjonarnych.

W początkowej części pracy omówiono aktualny stan wiedzy dotyczący procesów robotyzacji systemów produkcyjnych. Scharakteryzowano wymagania i ograniczenia, jakie on determinuje w elastycznych systemach technologicznych. Zaprezentowano aktualny stan wiedzy z zakresu metod planowania bezkolizyjnych trajektorii robotów manipulacyjnych. Omówiono podstawowy problem tego procesu, jakim jest analiza przestrzeni 3D otaczającej robota. Zaprezentowano istniejące metody, sklasyfikowano je i określono ich właściwości, w tym konieczność stosowania mechanizmów przeszukiwania heurystycznego. Rozważania te umożliwiły zdefiniowanie szczegółowego obszaru badań i wysunąć tezę prezentowanej pracy.

W zasadniczej swojej części praca obejmuje omówienie założeń i istoty zaproponowanej metody planowania ścieżek robotów manipulacyjnych. Omówiono podstawy przeprowadzonego procesu adaptacji metod planowania robotyki mobilnej, wyjaśniono i uzasadniono jego przyczyny. Wyposażając klasyczną metodę drzew oktalnych analizy przestrzeni w mechanizm uczenia się ze wzmocnieniem, określono podstawy opracowanego procesu planowania. Jego konsekwencją stało się wykorzystanie aparatu matematycznego procesów decyzyjnych Markowa, jako podstawy zastosowanego w metodzie procesu sterowania stochastycznego. Zdefiniowano podstawowe parametry omawianego procesu planowania, takie jak: warunki zajścia kolizji robota z otoczeniem, przestrzeń stanu robota, zbiór stanów sąsiednich, funkcje prawdopodobieństw warunkowych przejścia i osiągnięcia typowanych przez PLANER przemieszczeń robota. Całość metody uzupełniają opracowane dwie strategie lokalne pozwalające poddać prowadzony proces planowania optymalizacji.

Przestawiona metoda jest podstawą programu symulacyjnego Wiz_SRM będącego komputerowym modelem PLANERA reprezentującego robota podczas procesu planowania. Analiza uzyskanych wyników jednoznacznie wskazuje, że podstawowym walorem opracowanej metody jest prostota i efektywność czasowa dokonywanych analiz, będąca wynikiem wyeliminowania procesu poszukiwania heurystycznego rozwiązań dopuszczalnych. Ta cecha opracowanej metody stanowi jej podstawowy walor utylitarny.

Słowa kluczowe: planowanie trajektorii robotów manipulacyjnych, analiza przestrzeni, metody grafowe, procesy decyzyjne Markowa, funkcja wag, planowanie bezkolizyjnej ścieżki robota.

PLANNING COLLISION-FREE TRAJECTORIES OF MANIPULATIVE AND STATIONARY INDUSTRIAL ROBOTS ON THE BASIS OF MARCOV'S DECISION-MAKING PROCESSES AND EVALUATIVE FUNCTION

Abstract

The scope of the thesis is the presentation of several years' studies on the issue of planning collision-free trajectories of manipulative industrial robots. On the grounds of graphic methods that are widely popular in the field of robotics, an attempt was made to find a solution. This novel solution is based on adjusting highly efficient, in terms of time and calculation procedure, techniques of planning mobile robot motions to the task of planning the motion of industrial robots, both manipulative and stationary.

In the first part of the thesis, recent state of knowledge on the robotization of manufacturing systems is described. The requirements and constraints are characterized as determined for flexible technical systems. The update on most recent methods of planning collision-free trajectories of manipulative robots is provided. The basis problem of the analysis of the 3D space surrounding the robot is discussed. The existing methods are reviewed, classified and their properties determined, including the necessity of applying heuristic search methods. Accordingly, detailed scope of investigation is designated and the thesis put forward.

The main part of the thesis includes the discussion on the assumptions and the essence of the proposed method of planning collision-free trajectories of manipulative robots. The bases for the adjustment of mobile robotic planning are explained and substantiated. On the grounds of the classic octal trees method of analyzing space, enriched by the reinforced learning mechanism, the bases for the planning process are determined. Markov's decision making mathematical apparatus is applied to stochastic process control. The fundamental parameters of the process are defined, including: conditions of the collision between the robot and its surroundings, the space of the robot state, the set of contiguity states, conditional probability functions of robot passage and destinations as specified by the planner. The whole method is supplemented by two local strategies enabling the optimization of the planning process.

On the bases of the discussed method, Wiz_SRM simulation program was designed. The program is a computer model of the planner that represents the robot in the planning process. The results explicitly indicate that the basic advantage of the method, constituting its greatest practical strength, is the simplicity and efficiency of analyses, due to the elimination of the process of heuristic search for acceptable solutions.

Key words: *planning of manipulative robots trajectories, space analysis, graph methods, Markov's decision-making processes, function of weights, planning of collision-free trajectory*

WYDAWNICTWO POLITECHNIKI ŚLĄSKIEJ
ul. Akademicka 5, 44-100 Gliwice, tel. (0-32) 237-13-81
http://wydawnictwo.polsl.pl

Sprzedaż i Marketing
tel. (0-32) 237-18-48
wydawnictwo_mark@polsl.pl

Nakł. 100+50	Ark. wyd. 13	Ark. druk. 11,5	Papier offset. 70x100, 80 g
Oddano do druku 26.11.2004 r.		Podpisano do druku 26.11.2004 r.	Druk ukończ. w grudniu 2004 r.

Wydrukowano w Zakładzie Graficznym Politechniki Śląskiej w Gliwicach, ul. Kujawska 1
zam. 411/04

Książki Wydawnictwa można nabyć w księgarniach

GLIWICE

- Punkt Sprzedaży – ul. Akademicka 2 (237-17-87)
- „FORMAT" – Akademicka 5 (architektura i budownictwo)
- „LAMBDA" – ul. Akademicka 2 (237-21-40)
- Punkt Sprzedaży – ul. Akademicka 16 (automatyka, elektronika, informatyka)
- „ŻAK" – ul. Kaszubska

RYBNIK

- „ORBITA" – ul. Rynek 12
- „NEMEZIS" – ul. Hallera 26

ŁÓDŹ

- „POLITECHNIKA 100" – ul. Żeromskiego 116 PŁ.
- Hurtownia „BIBLIOFIL" – ul. Jędrowizna 9a (042) 679-26-77

KATOWICE

- Punkt Sprzedaży – ul. Krasińskiego 8
- Hurtownia „DIK" – ul. Dulęby 7 (032) 204-82-30
- Hurtownia „JERZY" – ul. Słoneczna 24 (258-99-58)

TYCHY

- „I Ja Tours" - ul. Piłsudskiego 10 (217-00-91 w.130)

ZABRZE

- Punkt Sprzedaży – ul. Roosevelta 26

KRAKÓW

- Techniczna – ul. Podwale 4 (012) 422-48-09
- Punkt Sprzedaży WND – AGH, Al. Mickiewicza 30

GDAŃSK

- EKO-BIS – ul. Dyrekcyjna 6 (058) 305-28-53

WARSZAWA

- Studencka – Pl. Politechniki 1 (022) 628-77-58
- Techniczna - ul. Kaliskiego 15 (022) 666-98-02
- Techniczna – ul. Świętokrzyska 14
- MDM – ul. Piękna 31

BIAŁYSTOK

- Dom Książki (Księgarnia 84) – ul. Wiejska 45 c

POZNAŃ

- Księgarnia „POLITECHNIK" – ul. Piotrowo 3 (061) 665-23-24
- Księgarnia Techniczna – ul. Półwiejska 28 (061) 659-00-38

NOWY SĄCZ

- Księgarnia „ATOM" – ul. Hoffmanowej 3 (018) 446-08-72

BIBLIOTEKA GŁÓWNA
Politechniki Śląskiej
P.3361/04/148

Druk: Drukarnia Gliwice, ul. Zwycięstwa 27, tel. 230 49 50

Wydawnictwo Politechniki Śląskiej
44-100 Gliwice, ul. Akademicka 5
tel./fax (0-32) 237-13-81
http:// wydawnictwo.polsl.pl
Sprzedaż i Marketing:
tel. (0 32) 237-18-48,
e-mail: wydawnictwo_mark@polsl.pl

BIBLIOTEKA GŁÓWNA
Politechniki Śląskiej
P.3361/04/148

Druk: Drukarnia Gliwice, ul. Zwycięstwa 27, tel. 230 49 50

Wydawnictwo Politechniki Śląskiej
44-100 Gliwice, ul. Akademicka 5
tel./fax (0-32) 237-13-81
http:// wydawnictwo.polsl.pl
Sprzedaż i Marketing:
tel. (0 32) 237-18-48,
e-mail: wydawnictwo_mark@polsl.pl