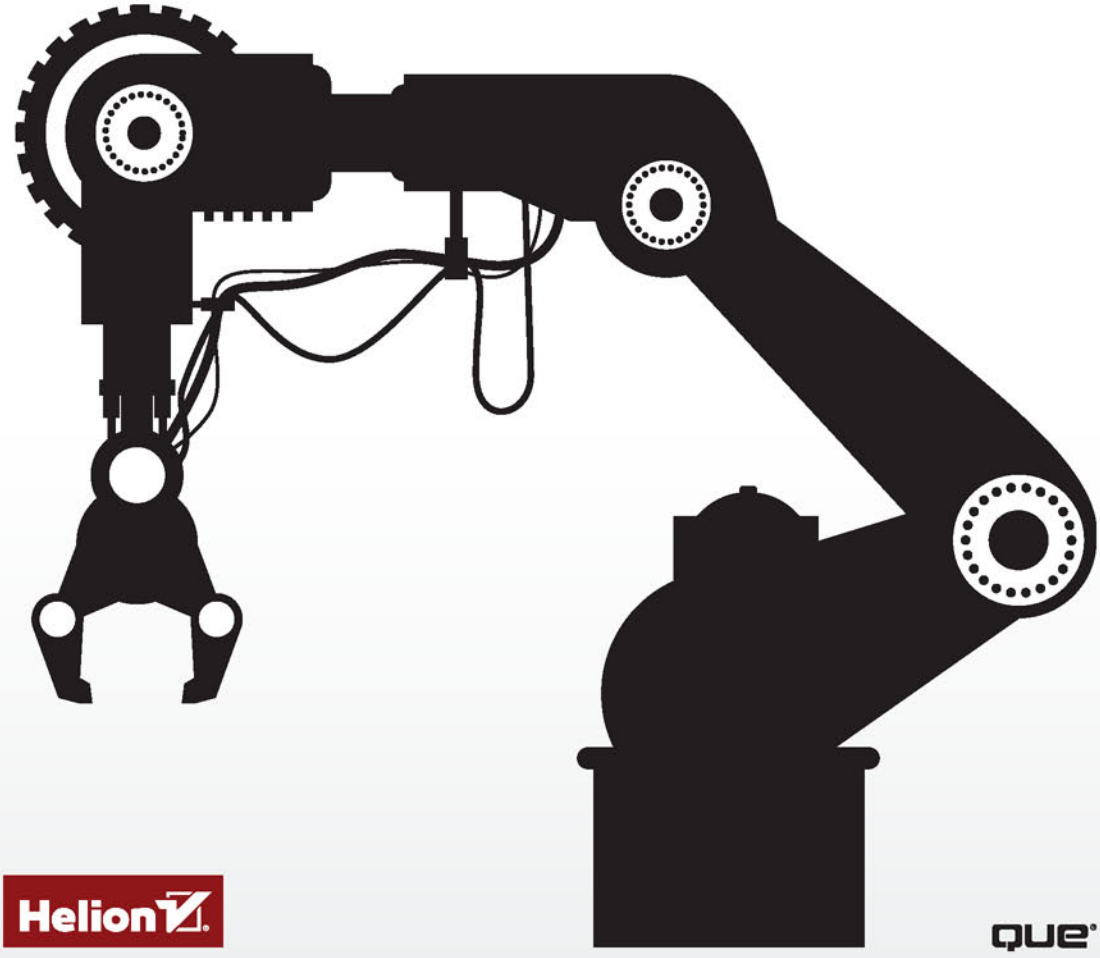
CAMERON HUGHES • TRACEY HUGHES

PROGRAMOWANIE ROBOTÓW

STEROWANIE PRACĄ ROBOTÓW AUTONOMICZNYCH



Helion

que

Tytuł oryginału: Robot Programming: A Guide to Controlling Autonomous Robots

Tłumaczenie: Konrad Matuk

ISBN: 978-83-283-2937-9

Authorized translation from the English language edition, entitled: ROBOT PROGRAMMING:
A GUIDE TO CONTROLLING AUTONOMOUS ROBOTS, ISBN 0789755009; by Cameron Hughes;
and by Tracey Hughes, published by Pearson Education, Inc, publishing as QUE Publishing.
Copyright © by 2016 by Pearson Education.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education Inc.
Polish language edition published by HELION SA. Copyright © 2017.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiejkolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie bierze jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Wydawnictwo HELION nie ponosi również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/prorob
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

• Kup książkę
• Poleć książkę
• Oceń książkę

• Księgarnia internetowa
• Lubię to! » Nasza społeczność

Spis treści

Kup książkę Poleć książkę

Kup książkę Poleć książkę

Kup książkę Poleć książkę

Kup książkę Poleć książkę

Kup książkę Poleć książkę

Kup książkę Poleć książkę

Wizualne planowanie scenariusza pracy robota

3

Trzecia lekcja robotyki: *Nie każ robotowi wykonywać zadania, którego nie potrafisz sobie wyobrazić.*

W rozdziale 2., „Słownictwo robotów", stwierdziliśmy, że roboty posługują się słownictwem — językiem umożliwiającym przekazanie im informacji na temat zadań, które mają wykonać w danej sytuacji lub w danym scenariuszu. Po określeniu słownictwa kolejnym etapem pracy jest tworzenie za jego pomocą instrukcji wykonywanych przez robota.

Zobrazowanie sobie, czyli stworzenie „wizualnej reprezentacji", scenariusza i instrukcji wykonywanych przez robota jest dobrym sposobem na zapewnienie prawidłowego ich wykonania przez robota. Obrazowa reprezentacja instrukcji pozwoli Ci przeanalizować kolejne ich kroki przed przekształceniem ich w kod programu. Dzięki tej technice, mając całościowy obraz scenariusza i możliwość dostrzeżenia ewentualnych problemów, lepiej zrozumiesz proces pracy robota i usprawnisz proces tworzenia kodu. Rozwiązanie takie określamy mianem wizualnego planowania scenariusza pracy robota. Proces ten pomaga w zaplanowaniu instrukcji, które robot ma wykonać. Wizualne planowanie pracy robota składa się z trzech elementów:

- plan obszaru, na którym ma pracować robot;
- tabela stanów robota i obiektów znajdujących się w obszarze jego pracy;
- schemat blokowy instrukcji związanych z zadaniami.

Te trzy elementy pozwolą uzyskać jasny obrazu czynności, która ma być wykonana przez program (może to być zapalenie świeczek na torcie albo nawet uratowanie świata przed zagładą). Elementy wizualnego planowania pracy robota mogą być używane w różnych kombinacjach. Schematy blokowe są dla niektórych bardziej czytelne od tabel stanów, a inni programiści wolą korzystać z tabel stanów. Niezależnie od tego, czy wolisz korzystać ze schematów blokowych czy tabel stanów, polecamy Ci jednoczesne posługiwanie się planem obszaru pracy robota.

Powiedzenie „jeden obraz jest wart więcej niż tysiąc słów" oznacza, że skomplikowaną myśl łatwiej jest wyrazić za pomocą obrazu niż długiego opisu słownego. Z tej koncepcji bardzo często korzystaliśmy w szkole podstawowej — rozwiązując liczne zadania, spotykaliśmy polecenie „narysuj rysunek", po którego wykonaniu rozwiązanie stawało się banalnie proste. Ta technika rozwiązywania problemów wciąż może nam być pomocna. Czasami narysowanie środowiska pracy robota, stworzenie tablicy stanów i opracowanie schematu blokowego pozwoli nam zaoszczędzić tysiąc słów, a także tysiąc poleceń. Dzięki wizualnemu planowi pracy robota zaplanujemy kolejne czynności wykonywane przez robota w różnych sytuacjach. W ten sposób nie będziesz musiał tworzyć kodu metodą prób i błędów.

Kup książkę Poleć książkę

Mapowanie scenariusza pracy robota

Pierwszym elementem wizualnego planowania pracy robota jest mapowanie scenariusza jego pracy. Mapa jest symbolicznym odwzorowaniem środowiska, w którym będą miały miejsce zadania i sytuacje. Środowisko danego scenariusza pracy jest światem, w którym będzie pracował Twój robot. Na rysunku 3.1 przedstawiliśmy klasyczną matę testową robota NXT Mindstorms.

Rysunek 3.1. Mata testowa robota NXT Mindstorms

Mata pokazana na rysunku 3.1 wchodzi w skład zestawu Mindstorms. Ma ona kształt prostokąta o wymiarach około 60×75 cm. Znajdziesz na niej 16 kolorów i 38 unikalnych liczb (niektóre z nich są zdublowane). Pole maty jest pokryte liniami prostymi i krzywymi. Na jej powierzchni nadrukowano żółte, niebieskie, czerwone i zielone kwadraty oraz inne kolorowe kształty. Mata ta tworzy środowisko przeznaczone do wstępnego testowania robotów NXT Mindstorms — umożliwia między innymi testowanie silników i czujników kolorów.

W podobny sposób (za pomocą różnych symboli) na mapę miejsca pracy robota nanosi się poszczególne przedmioty i przeszkody. Jeżeli przedmioty, z którymi robot ma nawiązać interakcję, są zbyt daleko lub wysoko, to czujniki robota mogą mieć problemy z ustaleniem ich położenia. Na mapie należy umieścić również wszystkie miejsca, do których według planu ma dotrzeć robot.

Wymiary przestrzeni, w której pracuje robot, a także wymiary samego robota, mogą wpływać na możliwość poruszania się robota i wykonania przez niego niektórych czynności. Wróćmy do przykładu robota BR-1. Jakie jest położenie robota względem tortu? Czy robot może do niego dotrzeć? Czy może napotkać na swej drodze jakieś przeszkody? Czy może poruszać się swobodnie po dostępnej przestrzeni? Stworzenie mapy przestrzeni, w której będzie pracował robot, ułatwi odpowiedzenie na te pytania.

Kup książkę Poleć książkę

WSKAZÓWKA

Najważniejszą rzeczą, poza samym robotem, na którą należy zwracać uwagę podczas programowania robota jest środowisko jego pracy.

Tworzenie planu miejsca pracy robota

Mapa może mieć formę prostego, dwuwymiarowego planu, na którym naniesione zostaną kształty, symbole i kolorowe linie symbolizujące różne przedmioty i inne roboty. W przypadku tak prostej mapy stosowanie dokładnej skali nie jest szczególnie ważne, ale warto zachować względne proporcje między wielkościami poszczególnych obiektów.

Wytycz obszar za pomocą prostych. Określ jednostki, w których będziesz podawać wymiary. Upewnij się, że korzystasz z tych samych jednostek, które zostały zastosowane w funkcjach interfejsu aplikacji. Odległości pokonywane przez robota, a także rozmiary różnych powierzchni i przedmiotów oznaczaj za pomocą strzałek i wartości liczbowych. Mapę najlepiej jest stworzyć w aplikacji przeznaczonej do obróbki grafiki wektorowej. Tworząc nasze mapy, posługiwaliśmy się programem Libre Office Draw. Na rysunku 3.2 pokazaliśmy prosty plan środowiska pracy robota BR-1.

Na rysunku zaznaczyliśmy wszystkie interesujące nas miejsca: położenie robota oraz stół i znajdujący się na nim tort. Ponadto nanieśliśmy wymiary obszaru roboczego, a także tor ruchu robota. Lewy dolny róg mapy oznaczyliśmy współrzędnymi (0,0), a prawy górny róg współrzędnymi (300,400). Są to wymiary obszaru wyrażone w centymetrach. Z mapy można odczytać również odległości pomiędzy przedmiotami a robotem. Plan ten nie jest stworzony w określonej skali, ale zachowaliśmy na nim proporcje między poszczególnymi wymiarami. Robot BR-1 ma 50 cm wysokości i 30 cm szerokości.

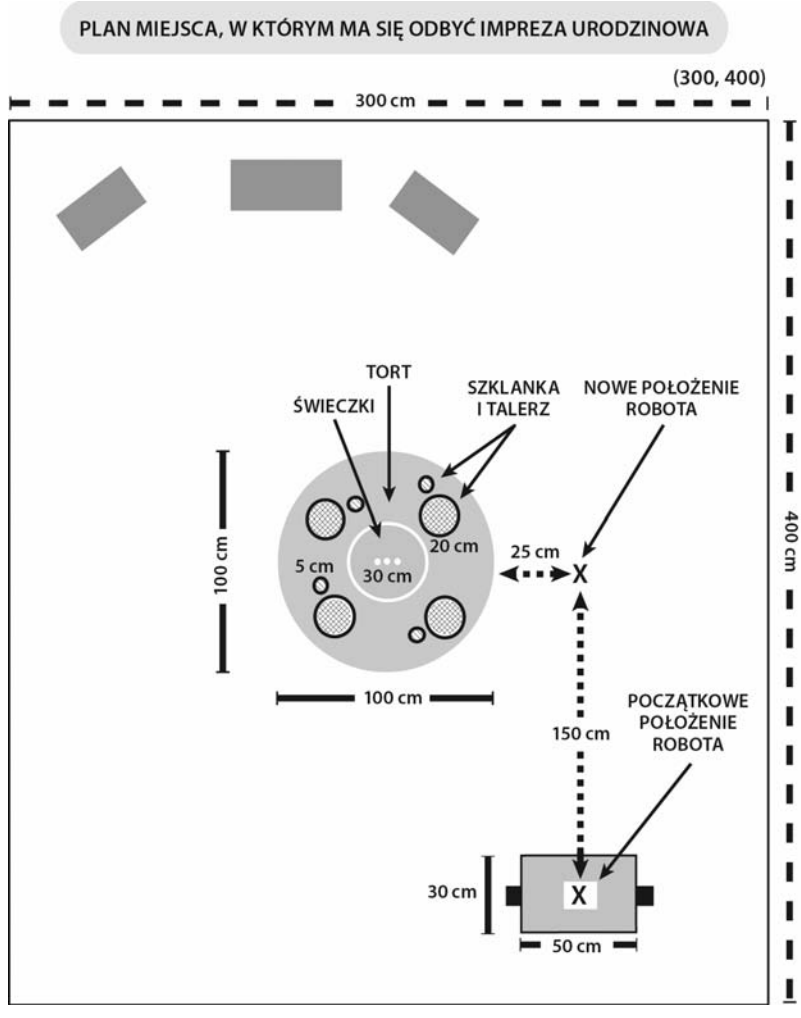
Robot ma zapalić świeczki na torcie, który znajduje się na środku obszaru o wymiarach 400×300 cm. Tort ma średnicę 30 cm i stoi na stole o wymiarach 100×100 cm. Dlatego ramię robota BR-1 powinno mieć długość 53 cm — tylko wtedy sięgnie ono z punktu *X* do najdalej od niego położonej świeczki.

Ramię robota może wysunąć zakończenie efektora końcowego na odległość 80 cm, a do wartości tej należy dodać jeszcze długość zapalniczki (10 cm). Planując zapalenie świeczek przez robota, należy jeszcze wziąć pod uwagę następujące czynniki:

- wysokość świeczki;
- wysokość tortu;
- odległość od ramienia robota do knota świeczki;
- miejsce, w którym znajduje się robot.

Na rysunku 3.3 pokazaliśmy sposób obliczania odległości, którą musi pokonać ramię robota, aby zapalić świeczkę. W tym przypadku odległość ta tworzy najdłuższy bok trójkąta. Odległość *a* (76 cm) jest różnicą między wysokością, na której znajduje się początek ramienia robota, a wysokością, na której znajduje się końcówka knota. Odległość *b* (53 cm) jest promieniem stołu, do którego dodajemy 3 cm (odległość między środkiem stołu a najdalszą świeczką).

Kup książkę Poleć książkę



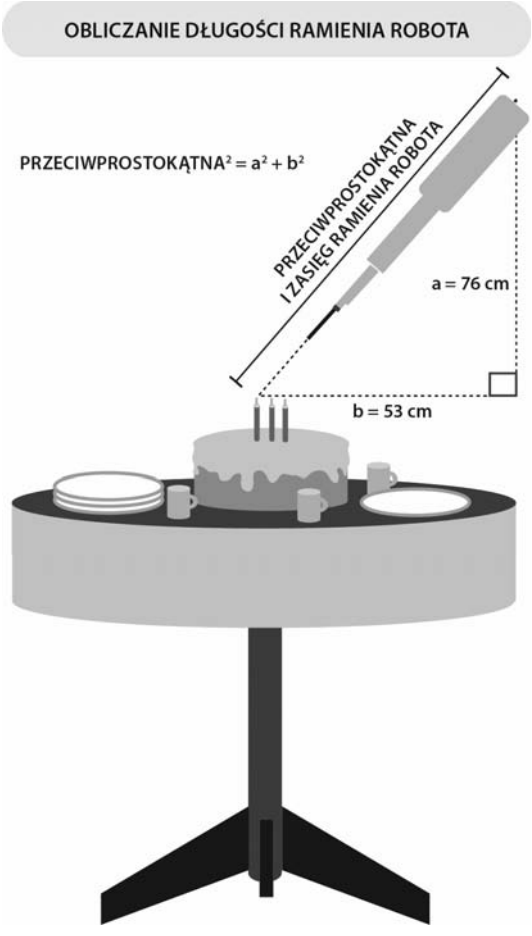
Rysunek 3.2. Plan środowiska pracy robota BR-1

W związku z tym zasięg ramienia, efektora końcowego i zapalniczki musi wynosić około 93 cm. Niestety robot dysponuje zasięgiem 90 cm, a więc będzie musiał jakoś pochylić się w stronę tortu, aby końcówka zapalniczki znalazła się 3 cm dalej i zapaliła najdalszą świeczkę.

UWAGA

Proces określania pozycji i zasięgu ramienia robota jest znacznie bardziej złożony od zaprezentowanego przykładu. Więcej informacji na ten temat znajdziesz w rozdziale 9., „Środowisko pracy robota". Przedstawiając ten przykład, chcieliśmy pokazać, jak plan miejsca pracy robota pomaga w rozwiązaniu ważnych problemów napotykanych podczas planowania zadań wykonywanych przez robota.

Kup książkę Poleć książkę



Rysunek 3.3. Obliczanie długości ramienia robota na podstawie długości przeciwprostokątnej

Otoczenie robota

UWAGA

Otoczeniem robota jest środowisko, w którym wykonuje on zadanie. Robot jest świadomy istnienia tylko tego środowiska. Na jego pracę nie wpływają żadne czynniki spoza tego środowiska i nie jest on świadomy istnienia niczego, co znajduje się poza jego otoczeniem.

Autonomiczny robot musi znać szczegóły otaczającego go środowiska. Na ile sprawnie udaje Ci się przemieszczać po mieście, którego nie znasz? Trudno Ci znaleźć miejsca, do których chcesz dotrzeć. Sprawę znacznie ułatwia mapa, na której jest zaznaczone, gdzie znajdują się na przykład restauracje i muzea. Robot, który jest *w pełni zautomatyzowany*, musi dysponować odpowiednią wiedzą na temat środowiska, w którym pracuje. Im więcej ma informacji, tym większe jest prawdopodobieństwo prawidłowego wykonania pracy.

Kup książkę Poleć książkę

Środowiska pracy różnią się od siebie, a niektóre z nich są dynamiczne. Nie wszystkie roboty mają dostęp do całości środowiska pracy. Pełna dostępność środowiska oznacza, że wszystkie obiekty i aspekty środowiska znajdują się w zasięgu czujników i efektorów robota, a więc żaden obiekt nie jest usytuowany zbyt wysoko, zbyt nisko lub zbyt daleko, aby robot mógł go wykryć lub nawiązać z nim interakcję, a ponadto robot jest wyposażony we wszystkie czujniki niezbędne do odbierania danych wejściowych dotyczących tego środowiska. Jeżeli w środowisku występuje dźwięk, to robot może go wykryć za pomocą czujnika dźwięku. Jeżeli w środowisku zostaje zapalone światło, to robot może je wykryć za pomocą czujnika światła.

Mianem częściowo dostępnego środowiska określamy środowisko, które nie może być w pełni eksplorowane przez robota, ponieważ nie jest on w stanie wykryć jego aspektów bądź znajdujących się w nim przedmiotów albo nawiązać z tymi przedmiotami interakcji, gdyż nie jest wyposażony w odpowiednie czujniki lub efektory końcowe. Przedmiot znajdujący się 180 cm nad ziemią jest poza zasięgiem robota o wysokości 50 cm wyposażonego w ramię o długości 80 cm. A gdyby robot BR-1 nie był wyposażony w czujnik dźwięku, a miałby zapalić świeczki po odśpiewaniu *Sto lat* przez gości? Dźwięk jest elementem środowiska, a więc robot nie mógłby wykonać zaplanowanej pracy. Dlatego tworząc plan miejsca pracy robota w środowisku, które nie jest w pełni dla niego dostępne, spójrz na nie z jego perspektywy. Umieszczając na planie przedmioty, które nie są dostępne dla robota, wyróżnij je jakiś symbolem — otocz je kolorową ciągłą lub przerywaną linią. Dzięki temu przedmioty te zostaną odróżnione od przedmiotów, do których robot ma dostęp.

Środowiska deterministyczne i niedeterministyczne

Co z kontrolą? Czy robot ma kontrolę nad każdym elementem środowiska, w którym się znajduje? Czy robot jest jedynym czynnikiem wpływającym na obiekty usytuowane w tym środowisku? To właśnie na tym polega różnica między środowiskiem **deterministycznym** i **niedeterministycznym**.

W środowisku deterministycznym kolejny stan zależy od bieżącego stanu i czynności wykonywanych przez robota (roboty). Oznacza to, że świeczki zapalone przez robota BR-1 będą się paliły, aż robot BR-1 ich nie zgasi. Jeżeli robot BR-1 ma pozbierać naczynia ze stołu, to goście nie powinni ich przenosić w inne miejsca.

W przypadku niedeterministycznych scenariuszy takich jak impreza urodzinowa robot BR-1 nie zdmuchuje świec (gdyby to zrobił, to wykazałby się dużą złośliwością). Naczynia mogą być przenoszone przez gości, a nie tylko przez robota BR-1. Co, jeżeli pomiędzy robotem BR-1 a celem, do którego zmierza, początkowo nie znajdowała się żadna przeszkoda, a nagle na jego drodze stanie gość? Jak robot ma pracować w dynamicznym, niedeterministycznym środowisku?

Praca w każdym środowisku wymaga rozwiązania określonych problemów. Robot pracujący w dynamicznym, niedeterministycznym środowisku musi dokonać analizy poprzedniego i bieżącego stanu, a dopiero później podejmować próbę wykonania zadania i określać, czy zadanie w ogóle może być wykonane.

W tabeli 3.1 przedstawiliśmy wybrane typy środowisk i ich krótkie charakterystyki.

Kup książkę Poleć książkę

Tabela 3.1. Wybrane typy środowisk i ich krótkie charakterystyki

Typ środowiska	Opis
W pełni dostępne	Wszystkie aspekty środowiska są dostępne dla czujników, siłowników i efektorów końcowych robota.
Częściowo dostępne	Niektóre przedmioty nie są dostępne dla robota lub nie mogą zostać wykryte przez jego czujniki.
Deterministyczne	Kolejny stan środowiska jest zależny tylko i wyłącznie od obecnego stanu i czynności wykonanych przez robota.
Niedeterministyczne	Kolejny stan środowiska nie zależy tylko i wyłącznie od działań robota; na przedmioty mogą wpływać również czynniki zewnętrzne.

Opis atrybutów środowiska pracy robota

UWAGA

Opis środowiskowych atrybutów robota jest listą zawierającą obiekty, które robot napotka na swej drodze, i obiekty, z którymi nawiąże interakcję. Ponadto zawiera on charakterystyki i atrybuty obiektów, które mogą zostać wykryte przez czujniki robota. Opis powinien zawierać również listę charakterystyk obiektów, które wpływają na sposób interakcji z nimi.

Wiele aspektów środowiska nie jest widocznych na mapie, ale powinny zostać gdzieś odnotowane w celu ich uwzględnienia podczas tworzenia instrukcji wykonywania zadań. Mogą to być dowolne czynniki środowiskowe, parametry odbierane przez czujniki lub właściwości wpływające na pracę silników i efektorów końcowych, takie jak kolor, masa, rodzaj powierzchni i siły zewnętrzne.

Niektóre z tych aspektów można nanieść na plan środowiska pracy robota, ale najlepiej jest stworzyć dodatkowy opis środowiskowych atrybutów robota zawierający tego typu informacje.

Kolor jest przykładem informacji odbieranej za pomocą czujnika koloru lub światła. Od masy przedmiotu zależy, czy robot może go podnieść (roboty są wyposażone w silniki o określonej mocy i momencie obrotowym). Możliwość manipulowania przedmiotem za pomocą efektorów końcowych zależy od kształtu, wysokości i powierzchni danego przedmiotu.

Każdy element charakteryzujący środowisko wchodzi w skład opisu środowiskowych atrybutów robota. Są to cechy dotyczące na przykład wymiarów, natężenia światła i powierzchni. Wpływają one na pracę czujników i silników. Rodzaj oświetlenia (światło słoneczne, fluorescencyjne lub żarowe) ma wpływ na odbiór kolorów przez czujniki. Robot poruszający się po drewnianej podłodze pracuje inaczej niż robot poruszający się po żużlu, piachu czy dywanie. Od powierzchni, po której robot się przemieszcza, zależy szybkość obrotu kół i przebyty dystans.

W tabeli 3.2 przedstawiliśmy opis środowiskowych atrybutów pracy robota na macie testowej Mindstorms NXT.

Kup książkę Poleć książkę

Tabela 3.2. Opis środowiskowych atrybutów pracy robota na macie testowej Mindstorms NXT

Przedmiot: fizyczna przestrzeń pracy

Atrybut	Wartość
Rodzaj środowiska	Deterministyczne, w pełni dostępne
Szerokość	60 cm
Długość	75 cm
Wysokość	0
Kształt	Prostokąt
Powierzchnia	Papier (gładka)

Przedmiot: kolor (światło)

Atrybut	Wartość
Liczba kolorów	16
Natężenie światła	16
Kolory	Czerwony, zielony, niebieski, żółty, pomarańczowy, biały, czarny, szary, jasnoniebieski, srebrny itd.

Przedmiot: symbole

Atrybut	Wartość
Symbol	Liczby całkowite
Wartości całkowitoliczbowe	0 – 30, 90, 120, 180, 270, 360, 40, 60, 70
Geometria	Linie, łuki, kwadraty

Opis środowiskowych atrybutów pracy robota na macie testowej określa środowisko pracy roboty — informuje o jego typie (w pełni dostępne i deterministyczne), wszystkich kolorach i symbolach. Ponadto zawiera on informacje o elementach napotykanych przez robota na przykład podczas poszukiwania niebieskiego kwadratu. Zaprezentowany opis zawiera listę atrybutów i wartości fizycznej przestrzeni roboczej, a także kolorów i symboli znajdujących się na macie testowej.

W przypadku dynamicznych środowisk takich jak opisana przez nas impreza urodzinowa opis atrybutów pracy robota może zawierać dodatkowo informacje dotyczące zewnętrznych sił wpływających na różne obiekty. Na przykład talerze i szklanki znajdują się na stole w swoich położeniach początkowych, ale po upływie pewnego czasu mogą został przełożone w inne miejsca stołu przez uczestników imprezy. Nowe współrzędne określające położenia naczyń powinny być umieszczone w opisie środowiskowych atrybutów pracy robota wraz z informacją o czasie, w którym doszło do ich przemieszczenia. Położenie wszystkich naczyń powinno być stale aktualizowane — tylko wtedy robot BR-1 będzie mógł je zebrać po zakończeniu imprezy. W tabeli 3.3 przedstawiliśmy opis środowiskowych atrybutów pracy robota BR-1 podczas imprezy urodzinowej.

Kup książkę Poleć książkę

Tabela 3.3. Opis środowiskowych atrybutów pracy robota BR-1 podczas imprezy urodzinowej

Przedmiot: fizyczna przestrzeń pracy

Atrybut	Wartość	Siła	Czas lub warunek	Nowa wartość
Rodzaj środowiska	Częściowo niedeterministyczne			
Szerokość	300 cm			
Długość	400 cm			
Wysokość	0			
Kształt	Prostokąt			
Powierzchnia	Papier (gładka)			
Oświetlenie	Sztuczne			

Przedmiot: tort

Atrybut	Wartość	Siła	Czas lub warunek	Nowa wartość
Wysokość	14 cm			
Średnica	30 cm			
Współrzędne	150, 200	Zewnętrzna	Nie dotyczy	
Położenie	Stół	Zewnętrzna	Nie dotyczy	
Powiązane przedmioty	Świeczki			

Przedmiot: świeczki

Atrybut	Wartość	Siła	Czas lub warunek	Nowa wartość
Wysokość	4 cm			
Liczba	3			
Współrzędne	1. 153, 200 2. 150, 200 3. 147, 200	Zewnętrzna	Nie dotyczy	
Warunek 1.	Zgaszone	BR-1	Rozpoczęcie śpiewu	Zapalone
Warunek 2.	Zapalone	Zewnętrzna	Zakończenie śpiewu	Zgaszone

Kup książkę Poleć książkę

Tabela 3.3. Opis środowiskowych atrybutów pracy robota BR-1 podczas imprezy urodzinowej — *ciąg dalszy*

Przedmiot: talerze

Atrybut	Wartość	Siła	Czas lub warunek	Nowa wartość
Średnica	20 cm			
Wysokość	1 cm			
Liczba	4			
Współrzędne	1. 110, 215 2. 110, 180 3. 170, 215 4. 170, 180	Zewnętrzna	Po zakończeniu przyjęcia	Wszystkie ułożone w stosie w punkcie o współrzędnych 110, 215 Wysokość: 2 cm

Przedmiot: szklanki

Atrybut	Wartość	Siła	Czas lub warunek	Nowa wartość
Średnica	5 cm			
Wysokość	10 cm			
Liczba	4			
Współrzędne	1. 119, 218 2. 105, 189 3. 165, 224 4. 163, 185	Zewnętrzna	Po zakończeniu przyjęcia	Wszystkie ułożone w stosie w punkcie o współrzędnych 119, 218 Wysokość: 14 cm

W tabeli opisu środowiskowych atrybutów robota pojawiły się trzy dodatkowe kolumny:

- Siła,
- Czas lub warunek,
- Nowa wartość.

Siła jest czynnikiem powodującym iterację obiektu; siłą może być dowolny czynnik środowiskowy niezależny od robota. **Czas** lub **warunek** informuje o tym, kiedy dochodzi do interakcji siły i obiektu. **Nowa wartość** mówi sama za siebie.

Wizualne planowanie scenariusza pracy robota za pomocą pseudokodu i schematu blokowego

Tworzenie schematu blokowego ma na celu określenie przepływu sterowania. Schemat taki ma postać sekwencji instrukcji i łączących je linii, które mogą zawierać pętle, a także operacje podejmowania decyzji i wyboru. Schemat blokowy opisuje proces za pomocą specjalnych pól symbolizujących określone czynności. Tekst wyświetlany w tych polach opisuje zadanie, proces lub instrukcję.

Kup książkę Poleć książkę

Schemat blokowy jest rodzajem diagramu stanów (diagramy te omawiamy w dalszej części tego rozdziału) — zawiera stany, które są zamieniane na czynności i działania. Schematy blokowe umożliwiają łatwe zilustrowanie procesów decyzyjnych i powtarzanie czynności, a także przedstawienie skutków tych działań. Niektórzy polecają tworzenie schematów blokowych przed napisaniem pseudokodu. Pseudokod można łatwo zamienić na kod programu lub umieścić w jego dokumentacji. Ponadto można go bez trudu zmodyfikować. Schemat blokowy trudniej zmodyfikować i wymaga zastosowania specjalnego edytora graficznego.

W tabeli 3.4 porównaliśmy wady i zalety pseudokodu i schematu blokowego. Oba narzędzia doskonale nadają się do określania kolejnych kroków pracy robota. Wybór narzędzia zależy od preferencji programisty i projektu, nad którym pracuje.

Tabela 3.4. Porównanie zalet i wad pseudokodu i schematu blokowego

Technika planowania scenariusza pracy robota	Zalety	Wady
Pseudokod: Metoda opisu instrukcji wykonywanych przez komputer wykorzystująca elementy języka naturalnego i języka programowania	Łatwy do utworzenia i zmodyfikowania za pomocą dowolnego edytora tekstu Implementacja jest przydatna w każdym projekcie Łatwy do napisania i zrozumienia Łatwy do przekształcenia na język programowania	Nie jest to technika wizualna Brak standardowego stylu lub formatu Logika jest trudniejsza do prześledzenia
Schemat blokowy: Tworzy się go od góry do dołu strony. Każde polecenie jest umieszczone w polu o określonym kształcie, a kierunek przepływu programu określają strzałki	Technika wizualna, ułatwia przekazanie koncepcji innym Umożliwia bardziej efektywną analizę problemów	Schemat przedstawiający złożone operacje logiczne może stać się duży i nieczytelny Wprowadzanie zmian może wiązać się z koniecznością rysowania schematu od początku

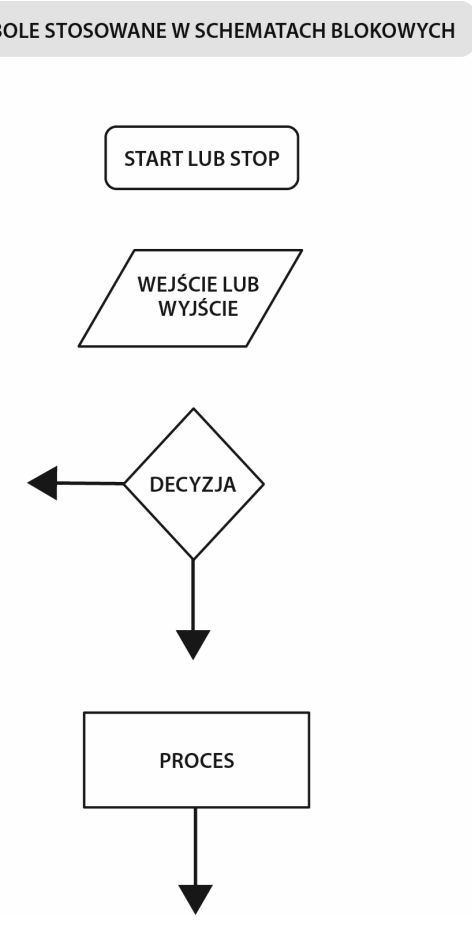
W schematach blokowych najczęściej spotyka się cztery rodzaje symboli:

- **Start i stop:** Symbol „start” (pole z etykietą „start”) oznacza początek schematu blokowego, a symbol „stop” (pole z etykietą „stop”) sygnalizuje jego koniec. To jedyne symbole oznaczone słownymi etykietami.
- **Wejście i wyjście:** Symbole wejścia i wyjścia zawierają dane wejściowe (np. dane dostarczane przez użytkownika) i dane wyjściowe (dane uzyskane w wyniku przetworzenia danych wejściowych).
- **Decyzje:** Symbol decyzji zawiera pytanie, na które należy odpowiedzieć.
- **Proces:** Symbol procesu zawiera krótki (składający się z kilku słów) opis reguły lub wykonywanej operacji.

Rysunek 3.4 przedstawia symbole używane w schematach blokowych.

Kup książkę Poleć książkę

Rysunek 3.4.
Symbole stosowane w schematach blokowych



Każdy symbol połączony jest z innym symbolem za pomocą strzałki. Symbolowi „start" towarzyszy tylko jedna strzałka, która jest skierowana na zewnątrz, a symbolowi „stop" towarzyszy tylko jedna strzałka, która jest skierowana do tego symbolu. Symbol „start" oznacza początek schematu blokowego. Wewnątrz niego umieszczona jest etykieta z napisem „start". Symbol „stop" natomiast sygnalizuje koniec schematu blokowego. Wewnątrz niego umieszczona jest etykieta z napisem „stop".

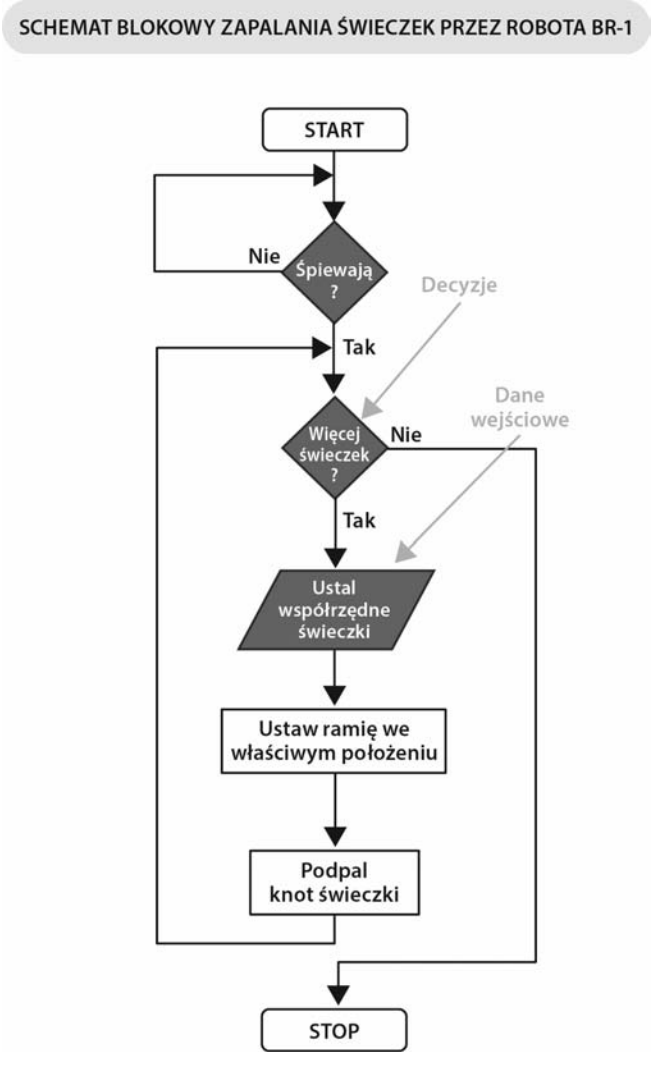
Symbole „start" i „stop" to jedyne znaki graficzne opatrzone słownymi etykietami. Symbol decyzji zawiera pytanie, na które program będzie musiał odpowiedzieć. Symbol procesu zawiera krótki (składający się z kilku słów) opis reguły lub wykonywanej operacji. Symbol decyzji styka się z trzema strzałkami: jedną skierowaną do wewnątrz i dwiema skierowanymi na zewnątrz. Każda z dwóch wychodzących strzałek określa ścieżkę procesu zależną od podjętej decyzji. Są one oznaczone etykietami:

- prawda (tak),
- fałsz (nie).

Kup książkę Poleć książkę

Symbol procesu i symbol wejścia lub wyjścia charakteryzują się jedną strzałką skierowaną do wewnątrz i jedną skierowaną na zewnątrz. Wewnątrz tych symboli zapisywane są informacje o regule, wykonywanej operacji i danych wejściowych lub wyjściowych. Na rysunku 3.5 zamieściliśmy schemat blokowy operacji zapalania świeczek.

Rysunek 3.5. Schemat blokowy operacji zapalania świeczek przez robota BR-1



Zauważ, że na początku schematu blokowego, pod symbolem „start", pokazany jest proces czekania robota BR-1 na rozpoczęcie śpiewania. Robot podejmuje decyzję określającą rozpoczęcie śpiewania. Może on podjąć jedną z dwóch decyzji: jeżeli na postawione pytanie pada odpowiedź „nie" (śpiewanie jeszcze się nie rozpoczęło), to robot czeka dalej, a jeżeli odpowiedź jest twierdząca (śpiewanie się rozpoczęło), to robot zaczyna wykonywanie pętli (podejmuje kolejną decyzję).

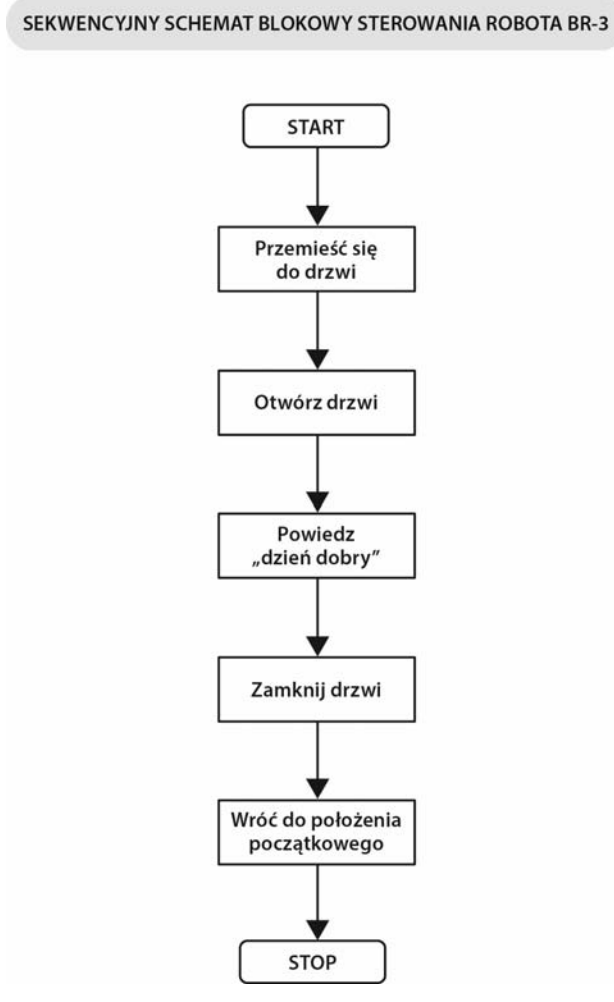
Kup książkę Poleć książkę

Jeżeli na torcie znajdują się jeszcze jakieś niezapalone świeczki, to robot ustala współrzędne kolejnej świeczki, ustawia ramię w odpowiednim położeniu, a potem podpala knot. Operacja ustalania współrzędnych świeczki została przedstawiona za pomocą symbolu danych wejściowych. Robot powinien zapalić wszystkie świeczki, a następnie zakończyć pracę.

Przepływ sterowania i struktury sterujące

Zadania wykonywane przez robota mogą być serią kolejnych czynności — sekwencyjnym przepływem sterowania. Termin **przepływ sterowania** oznacza kierunek przetwarzania instrukcji programu. Przepływ sterowania określa sposób, w jaki komputer reaguje na dane warunki i parametry. Rysunek 3.6 przedstawia przykład sekwencyjnego przepływu sterowania. BR-3 jest kolejnym robotem obecnym na naszej imprezie urodzinowej. Ma on za zadanie otwierać drzwi gościom. Rysunek 3.6 przedstawia sekwencyjny przepływ sterowania robota BR-3.

Rysunek 3.6. Schemat blokowy sterowania robota BR-3

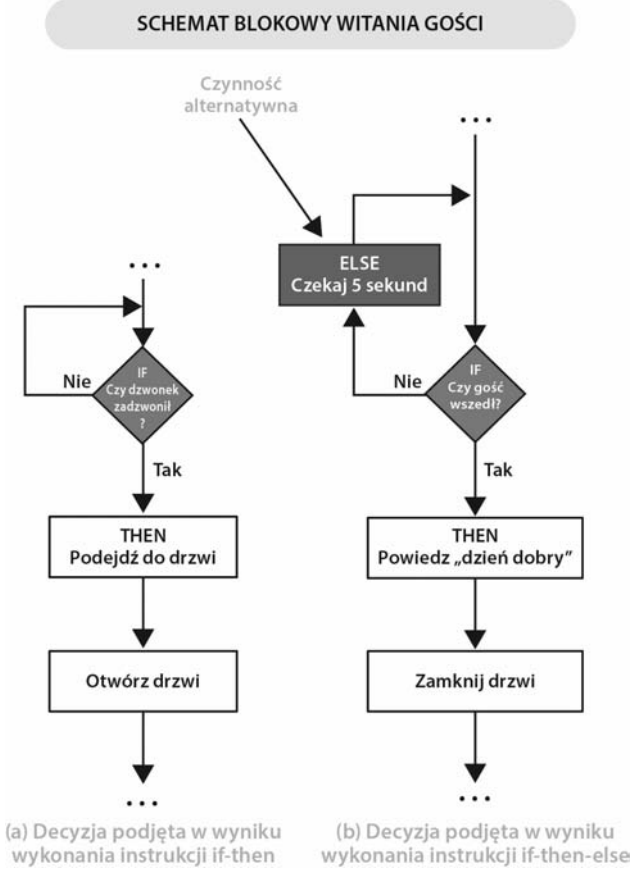


Kup książkę Poleć książkę

Robot podchodzi do drzwi, otwiera je, mówi „dzień dobry", a następnie zamyka drzwi i wraca do miejsca, z którego ruszył. Zachowuje się w sposób nieco nieprzemyślany. Czy obecność gościa pod drzwiami została zasygnalizowana dźwiękiem dzwonka? Czy po wypowiedzeniu słów „dzień dobry" robot pozwolił wejść gościowi do środka przed zamknięciem drzwi? Robot BR-3 powinien działać w sposób przewidywalny. Powinien on podejmować decyzje w sposób powtarzalny, w wyniku zajścia określonych zdarzeń.

Symbol decyzji jest używany do konstruowania rozgałęzień alternatywnych przepływów sterowania programu. Symbole te mogą być stosowane do wyrażania decyzji, powtórzeń i instrukcji warunkowych. Prosta decyzja ma strukturę instrukcji `if-then` (jeżeli, to) lub `if-then-else` (jeżeli, to w przeciwnym wypadku).

Rysunek 3.7(a) przedstawia prostą instrukcję `if-then` opisującą proces podejmowania decyzji przez robota BR-3: „Jeżeli (`if`) zadzwoni dzwonek, to (`then`) podejdź do drzwi i je otwórz". Teraz robot BR-3 będzie czekał, aż gość wejdzie, i dopiero wtedy powie „dzień dobry". Zwróć uwagę na alternatywną czynność, która jest wykonywana w przypadku, gdy gość jeszcze nie wszedł do mieszkania: robot BR-3 poczeka 5 sekund, a następnie sprawdzi ponownie, czy gość już wszedł. Jeżeli gość wszedł do środka, to robot wita go słowami „dzień dobry" i zamyka drzwi. Jest to instrukcja `if-then-else`, którą pokazaliśmy na rysunku 3.7(b) — alternatywną czynnością jest czekanie.



Rysunek 3.7. Schemat blokowy instrukcji if-then i if-then-else

Kup książkę Poleć książkę

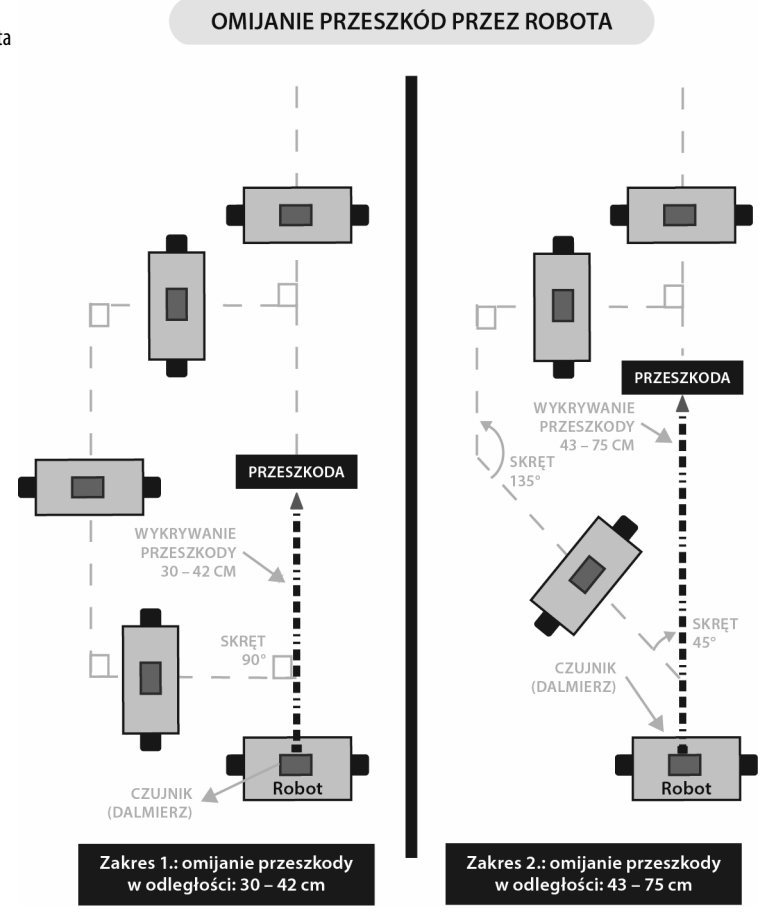
Instrukcja pokazana na rysunku 3.7 sprawdza warunek zadzwonienia do drzwi. A gdybyśmy chcieli sprawdzić więcej warunków przed otwarciem drzwi przez robota BR-3? Robot BR-1 powinien sprawdzić kilka warunków przed zapaleniem świec:

- „Jeżeli (`if`) słychać śpiew i (`and`) zapalniczka jest zapalona, to (`then`) zapal świece."

W tym przypadku oba warunki muszą być spełnione. Taką konstrukcję nazywamy warunkiem **zagnieżdżonym**.

A co, jeśli mamy do czynienia z pytaniem (warunkiem), na które jest kilka możliwych odpowiedzi, a każda odpowiedź ma skutkować wykonaniem innej czynności? Na przykład BR-1 lub BR-3 na swojej drodze spotka przeszkodę i musi ją obejść, aby dotrzeć do celu. W takiej sytuacji robot może sprawdzić odległość przeszkody i określić czynność, którą należy wykonać w celu jej ominięcia. Jeżeli przeszkoda znajdzie się w zasięgu działania czujników, to robot może skręcić w lewo o 90° lub 45°, obejść przeszkodę, a następnie wrócić na standardowy tor ruchu i podążać dalej w kierunku celu (zobacz rysunek 3.8).

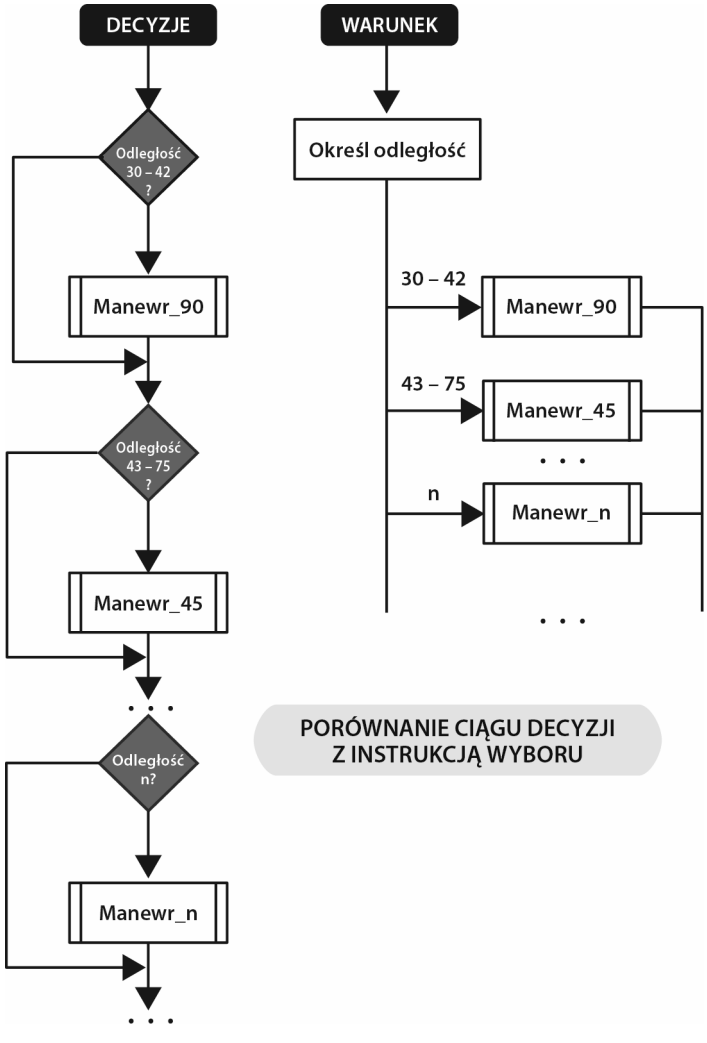
Rysunek 3.8.
Omijanie przeszkód przez robota



Kup książkę Poleć książkę

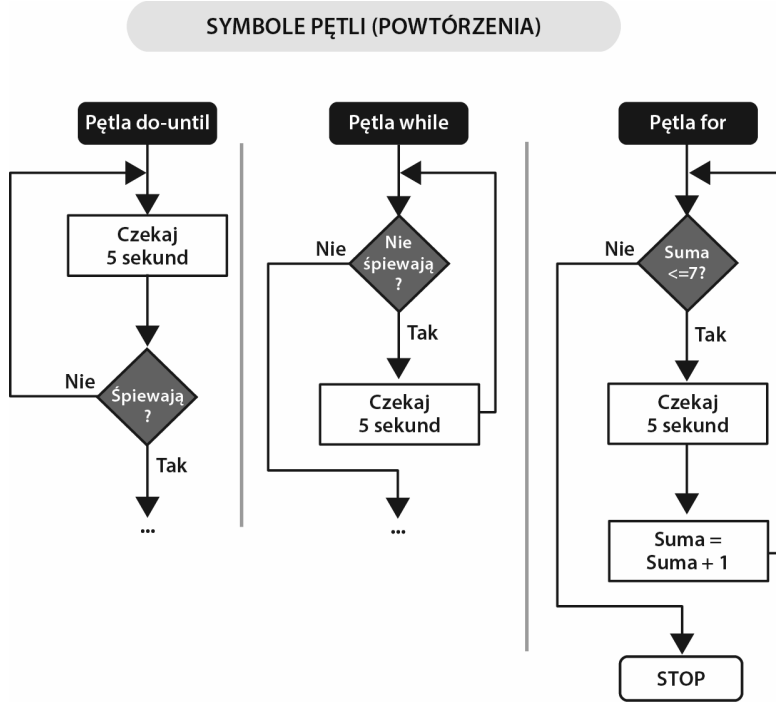
Można to wyrazić na schemacie blokowym jako ciąg instrukcji wyboru. Instrukcja wyboru wymaga od robota podjęcia decyzji polegającej na odpowiedzeniu na pytanie, na które może istnieć kilka różnych odpowiedzi. Podczas podejmowania ciągu decyzji dochodzi do trzykrotnego zadania tego samego pytania, na które za każdym razem uzyskiwana jest inna odpowiedź, a robot za każdym razem wykonuje inną czynność. W przypadku instrukcji wyboru pytanie jest zadawane tylko raz. Na rysunku 3.9 został porównany ciąg decyzji z instrukcją warunkową, która sprawia, że cały proces jest bardziej czytelny i zrozumiały.

Rysunek 3.9. Porównanie ciągu decyzji z instrukcją wyboru



Na rysunku 3.10 przedstawiliśmy powtarzanie (zapętlenie) instrukcji. W pętli sprawdzany jest prosty warunek, a następnie wykonywana jest jakaś czynność. W zależność od jej rezultatu może ona być wykonana ponownie. Warunek może być sprawdzany przed wykonaniem czynności lub po. Rysunek 3.10(a) przedstawia czynność, która zostanie wykonana przynajmniej raz.

Kup książkę Poleć książkę



Rysunek 3.10. Schematy blokowe pętli (a) do-until, (b) while i (c) for

Jeżeli warunek nie zostanie spełniony (śpiewanie jeszcze się nie rozpocznie, bo może wszyscy zbyt dobrze się bawią), robot będzie dalej czekał. Jest to przykład pętli `do-until` (wykonuj (czynność) aż do (spełnienia warunku)). Pętla `while` (podczas gdy) najpierw sprawdza warunek, a następnie wykonuje czynność. Widać to na rysunku 3.10(b) — robot będzie czekał, jeśli śpiewanie jeszcze się nie rozpoczęło. Robot BR-1 będzie wykonywał pętlę i czekał na rozpoczęcie śpiewów podobnie jak w przypadku pętli `do-until`. Różnica polega na tym, że czynność czekania jest wykonywana po sprawdzeniu warunku. Na rysunku 3.10(c) pokazaliśmy kolejną pętlę, `for`, w której warunek jest sprawdzany określoną liczbę razy podczas jej wykonywania.

Podprocedury

Określając rolę odgrywaną przez robota w danym scenariuszu, można ją rozbić na ciąg czynności. Robot BR-1 ma odgrywać rolę gospodarza, którą można podzielić na następujące stany:

- bezczynność,
- przemieszczanie się,
- zapalanie świeczek,
- czekanie,
- zbieranie naczyń.

Kup książkę Poleć książkę

Rolę tę można podzielić również na ciąg czynności lub zadań:

1. Poczekaj na rozpoczęcie śpiewania.
   - Przejdź do stołu z tortem.
   - Zapal świeczki na torcie.
   - Wróć do punktu wyjścia.
2. Poczekaj na zakończenie imprezy.
   - Sprzątnij naczynia ze stołu.
   - Wróć do punktu wyjścia.

Powyższe krótkie opisy zadań można podzielić na sekwencje kroków (podprocedury). „Zapalanie świeczki" to złożony stan, który można podzielić na dwa podstany:

- lokalizowanie knota,
- podpalanie knota.

Tak naprawdę czynność „sprzątnij naczynia ze stołu" i „wróć do punktu wyjścia" również należałoby podzielić na podprocedury. Pozbieranie talerzy i szklanek ze stołu wymaga właściwego wykonania podprocedur ustawienia ramienia robota dla każdego zabieranego naczynia, a przemieszczanie się wymaga wykonywania podprocedur polegających na sterowaniu pracą silników.

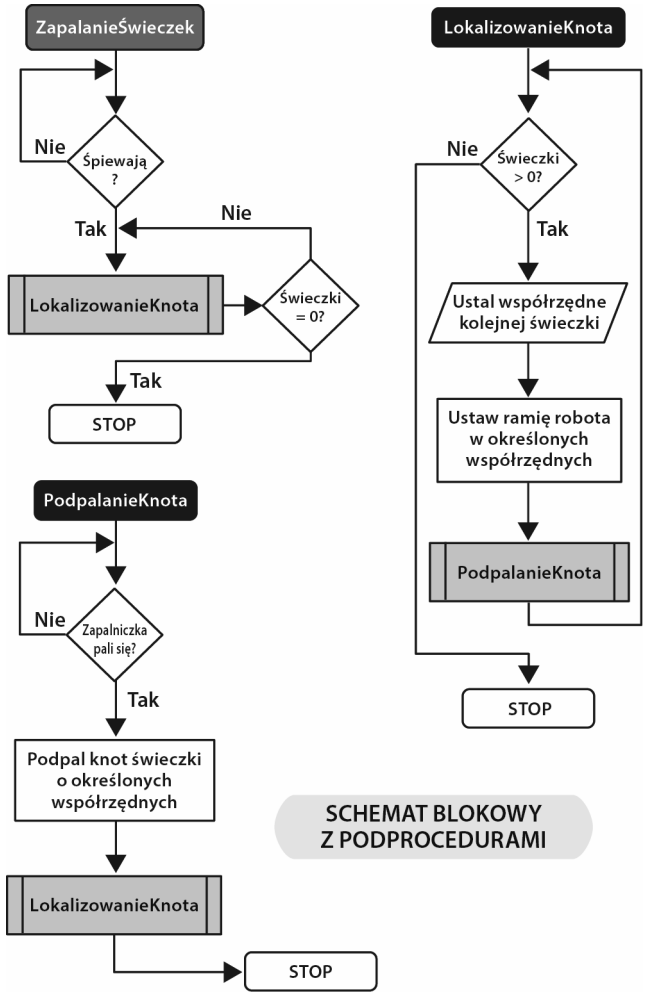
Na rysunku 3.11 pokazaliśmy schemat blokowy czynności `ZapalanieŚwieczek` i jej podprocedur — `LokalizowanieKnota` i `PodpalanieKnota`.

Podprocedura oznaczana jest za pomocą tego samego symbolu co proces, ale w symbolu tym umieszczana jest nazwa podprocedury otoczona po obu bokach pionowymi liniami. Nazwą podprocedury może być fraza, która opisuje jej działanie.

Po określeniu nazw procedur możliwe staje się stworzenie schematów blokowych. Korzystanie z podprocedur jest dobrym rozwiązaniem, ponieważ nie zmusza do natychmiastowego definiowania szczegółów tych operacji. Określenie dokładnego sposobu wykonywania zadań może być na razie bardzo trudne. Najpierw możemy opracować sekwencję procesów wyższego poziomu, a później podzielić je na czynności podrzędne.

Podprocedurę można określić i uogólnić na podstawie podobnych kroków wykonywanych w różnych momentach pracy robota. Zamiast powtarzać sekwencję kroków lub opracowywać różne podprocedury proces można uogólnić i umieścić w jednej podprocedurze, która będzie wywoływana w dowolnym momencie. Na przykład procedura przemieszczania się może być podzielona na sekwencję kroków pozwalających robotowi BR-1 pokonać drogę do stołu z tortem (`StółChód`) i sekwencję kroków umożliwiających mu powrót do punktu wyjścia (`PowrótChód`). Są to te same zadania, różniące się jedynie współrzędnymi punktu początkowego i końcowego. Zamiast podprocedur korzystających ze współrzędnych początkowych i końcowych możemy zastosować podprocedurę `Chód`, która wymaga bieżących i docelowych współrzędnych robota.

Kup książkę Poleć książkę



Rysunek 3.11. Schemat blokowy czynności ZapalanieŚwieczek i jej podprocedur — LokalizowanieKnota i PodpalanieKnota

Diagramy stanów robotów i obiektów

Diagram stanów jest jednym ze sposobów umożliwiających wizualizację maszyny stanów.

UWAGA

Maszyna stanów pozwala zaprezentować zachowanie pojedynczego robota lub obiektu znajdującego się w określonym środowisku. Stany są transformacjami robota lub obiektu, do których dochodzi w wyniku określonych zdarzeń.

Zmiana stanu może polegać na czymś tak prostym jak zmiana miejsca. Gdy robot przesuwa się z położenia początkowego do kolejnego miejsca docelowego, wówczas następuje zmiana stanu robota. Innym przykładem jest zmiana stanu świeczek znajdujących się na torcie (mogą być zapalone lub niezapalone).

Kup książkę Poleć książkę

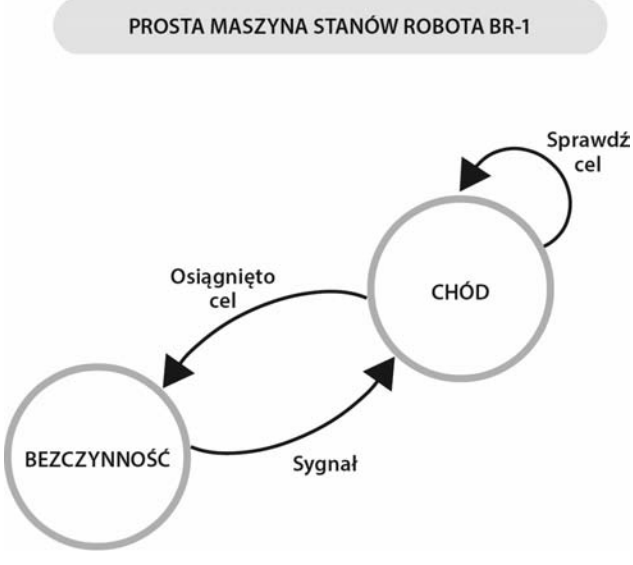
Maszyna stanów rejestruje zdarzenia, transformacje i reakcje. Diagram stanów przedstawia te aktywności. Jest on używany do przedstawiania wszystkich możliwych sytuacji, w których może się znaleźć obiekt w danym scenariuszu pracy robota. W rozdziale 2. pisaliśmy, że sytuację można porównać do stopklatki prezentującej pewne zdarzenie wyjęte ze scenariusza. Robot BR-1 może znaleźć się w następujących sytuacjach:

- **Sytuacja 1.**: Czeka na sygnał do przejścia w nowe miejsce.
- **Sytuacja 2.**: Przemieszcza się do stołu z tortem.
- **Sytuacja 3.**: Znajduje się przy stole z tortem, na którym znajdują się jeszcze nie zapalone świeczki.
- **Sytuacja 4.**: Pozycjonuje ramię nad świeczkami itd.

Wszystkie te sytuacje odzwierciedlają zmiany stanu robota. Zmiana stanu robota lub obiektu zachodzi wskutek jakiegoś zdarzenia. Do zdarzenia może dojść w następstwie sygnału, przeprowadzenia operacji lub upływu czasu. W wyniku zdarzenia następuje jakaś aktywność. Zależy ona od bieżącego stanu obiektu. Bieżący stan określa możliwości rozwoju sytuacji.

Zdarzenie wywołuje lub stymuluje warunek niezbędny do zmiany stanu. Zmianę stanu określamy mianem przejścia. Obiekt przechodzi ze stanu A (stanu źródłowego) do stanu B (stanu docelowego). Na rysunku 3.12 pokazaliśmy prostą maszynę stanów ilustrującą pracę robota BR-1.

Rysunek 3.12. Maszyna stanów ilustrująca pracę robota BR-1



Rysunek 3.12 przedstawia dwa stany robota BR-1: bezczynność i chód. Robot w stanie bezczynności czeka na zdarzenie, które stanowi sygnał konieczności przemieszczenia się w inne miejsce. Po otrzymaniu tego sygnału przechodzi ze stanu bezczynności w stan chodu i przemieszcza się do miejsca docelowego, a następnie przechodzi z powrotem w stan bezczynności. Sygnały, czynności i aktywności mogą być wykonywane lub kontrolowane przez obiekt bądź siły zewnętrzne. Na przykład nowa lokalizacja nie zostanie wygenerowana przez robota BR-1, lecz przez inny podmiot. Robot ten nie może sprawdzać swoich współrzędnych podczas wykonywania ruchu.

Kup książkę Poleć książkę

Tworzenie diagramu stanów

UWAGA

W przypadku diagramu stanów węzły są stanami, a linie przejściami. Stany są symbolizowane za pomocą okręgów lub prostokątów z zaokrąglonymi rogami, w których umieszcza się nazwę stanu. Przejścia to linie łączące stan źródłowy ze stanem docelowym. Strzałki zawsze wskazują stany docelowe.

Jak już pisaliśmy, stan jest warunkiem lub sytuacją i symbolizuje transformację obiektu. Maszyna stanów pokazuje stany i przejścia pomiędzy nimi. Można ją przedstawić na wiele sposobów — w tej książce jest to diagram UML (diagram sporządzony w zunifikowanym języku modelowania). Diagramy stanów umożliwiają zapis informacji dotyczących zdarzeń, czynności, warunków, elementów przejścia, a także rodzajów i elementów stanów.

Istnieją trzy typy stanów:

- **Stan początkowy** — jest to domyślny punkt początkowy maszyny stanów. Oznaczamy go czarną kropką, z której wyprowadzona jest strzałka do pierwszego stanu maszyny.
- **Stan końcowy** — jest to stan, w którym obiekt osiągnął koniec swojego cyklu życiowego. Symbolizujemy go za pomocą kropki umieszczonej wewnątrz okręgu.
- **Stan złożony i podstan** — jest to stan zawarty w innym stanie, określanym mianem stanu nadrzędnego lub stanu złożonego.

Stany składają się z różnych elementów. W tabeli 3.5 przedstawiliśmy elementy stanów opatrzone krótkimi opisami. Węzeł stanu, w którym znajduje się jego nazwa, może również zawierać elementy wymienione w tej tabeli. Mogą one być używane do opisu procesów, do których dochodzi podczas przejścia obiektu do nowego stanu. Mogą istnieć akcje, które muszą zostać wykonane od razu, gdy obiekt wejdzie w jakiś stan, lub bezpośrednio po jego wyjściu z tego stanu. Istnieją także akcje, które muszą zostać wykonane, gdy obiekt znajduje się w danym stanie. Wszystkie te informacje mogą być zawarte diagramie stanów.

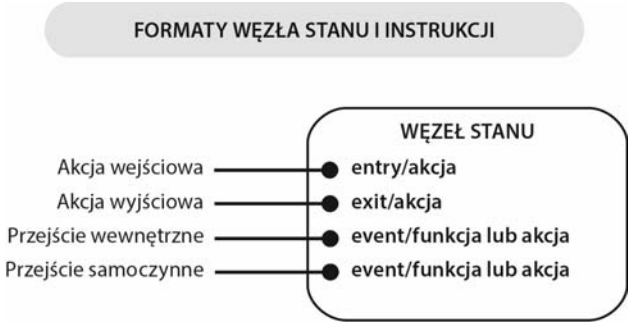
Tabela 3.5. Elementy stanu

Element	Opis
Nazwa	Unikalna nazwa odróżniająca dany stan od innych.
Akcje wejściowe i wyjściowe	Akcje podejmowane podczas wejścia w dany stan (akcje wejściowe) lub akcje podejmowane podczas wyjścia z danego stanu (akcje wyjściowe).
Stany złożone i podstany	Stan zagnieżdżony; podstany są stanami aktywowanymi w ramach stanu złożonego.
Przejścia wewnętrzne	Przejścia, do których dochodzi wewnątrz stanu. Są one obsługiwane bez wywoływania zmiany stanu. Nie powodują wykonania akcji wejściowych i wyjściowych.
Przejścia samoczynne	Przejścia, do których dochodzi wewnątrz stanu. Są one obsługiwane bez wywoływania zmiany stanu. Powodują wykonanie akcji wyjściowych i wejściowych podczas wychodzenia i ponownego wchodzenia w stan.

Kup książkę Poleć książkę

Na rysunku 3.13 został pokazany węzeł stanu i format zapisu deklaracji akcji, aktywności i przejść wewnętrznych.

Rysunek 3.13.
Węzeł stanu i format instrukcji



Instrukcje akcji wejściowych (`entry`) i wyjściowych (`exit`) są formatowane w następujący sposób:

- `entry/akcja lub aktywność`
- `exit/akcja lub aktywność`

Oto przykłady instrukcji akcji wejściowej i wyjściowej sprawdzania poprawności danych:

- akcja wejściowa: `entry/sprawdź(dane)`,
- akcja wyjściowa: `exit/wyślij(dane)`.

Po wejściu w stan sprawdzania poprawności danych wywoływana jest funkcja `sprawdź(dane)`. Przed wyjściem z tego stanu wywoływana jest funkcja `wyślij(dane)`.

W ramach stanu może dochodzić do przejść wewnętrznych — zdarzeń następujących po akcjach wejściowych, a przed akcjami wyjściowymi. Przejścia samoczynne to nie przejścia wewnętrzne. Wraz z przejściami samoczynnymi wykonywane są akcje wejściowe i wyjściowe. Przed wyjściem ze stanu wykonywana jest akcja wyjściowa.

Następnie dochodzi do ponownego wejścia w ten sam stan i ponownego wykonania akcji wejściowej. Akcja przejścia samoczynnego jest wykonywana po akcji wyjściowej, a przed akcją wejściową. Przejście samoczynne są przedstawiane jako linia, która po wyjściu ze stanu wraca na jego początek.

Instrukcje przejść wewnętrznych i przejść samoczynnych mają następujący format:

- `nazwa/akcja lub funkcja`

Na przykład:

- `do / utwórzTablicę(dane)`

Element `do` jest etykietą aktywności. W przytoczonym przykładzie uruchamiana jest funkcja `utwórzTablicę(dane)`.

Istnieje kilka elementów przejścia — zależności pomiędzy dwoma stanami. Przejścia wywoływane są przez wyzwalacze, z którymi związane mogą być jakieś akcje. Spełnienie jakiegoś warunku może również wywołać przejście. W tabeli 3.6 przedstawiliśmy elementy przejścia.

Kup książkę Poleć książkę

Tabela 3.6. Elementy przejścia

Element	Opis
Stan źródłowy	Pierwotny stan obiektu; w wyniku przejścia obiekt opuszcza ten stan.
Stan docelowy	Stan, w którym znajdzie się obiekt w wyniku przejścia.
Wyzwalacz zdarzenia	Zdarzenie, które doprowadza do przejścia. Do przejścia może dojść bez wyzwalacza — od razu po zakończeniu wszystkich aktywności stanu źródłowego.
Warunek wartowniczy	Warunek logiczny, którego spełnieniu dochodzi do przejścia.
Akcja	Akcja wykonywana przez obiekt podczas przejścia. Może ona być związana z wyzwalaczem zdarzenia lub warunkiem wartowniczym.

Wyzwalacz zdarzenia charakteryzuje się formatem podobnym do formatu instrukcji akcji stanu:

- `nazwa/akcja lub funkcja`,
- `nazwa [warunek wartowniczy] / akcja lub funkcja`.

Oto przykład dodawania warunku wartowniczego do instrukcji przejścia wewnętrznego:

- `do [Validated] / utwórzTablicę(dane)`

UWAGA

Warunek wartowniczy jest wartością logiczną lub wyrażeniem logicznym, które mogą dać wartość `true` (prawda) lub `false` (fałsz). Warunek ten jest umieszczany w nawiasach kwadratowych. Funkcja jest wywoływana tylko wtedy, gdy jest on spełniony. Może być stosowany w instrukcjach stanu lub przejścia.

3

UWAGA

`Validated` to wartość logiczna. Jest to warunek konieczny do wykonania funkcji `utwórzTablicę(dane)`.

Na rysunku 3.14 zamieściliśmy diagram stanów robota BR-1.

Na diagramie znajdują się cztery stany: bezczynność, chód, zapalanie świeczek i zbieranie naczyń. Przechodząc ze stanu bezczynności w stan chodu, robot ustala nową lokalizację i zna cele misji:

- `do [UstalPozycję] / wybierzMisję()`

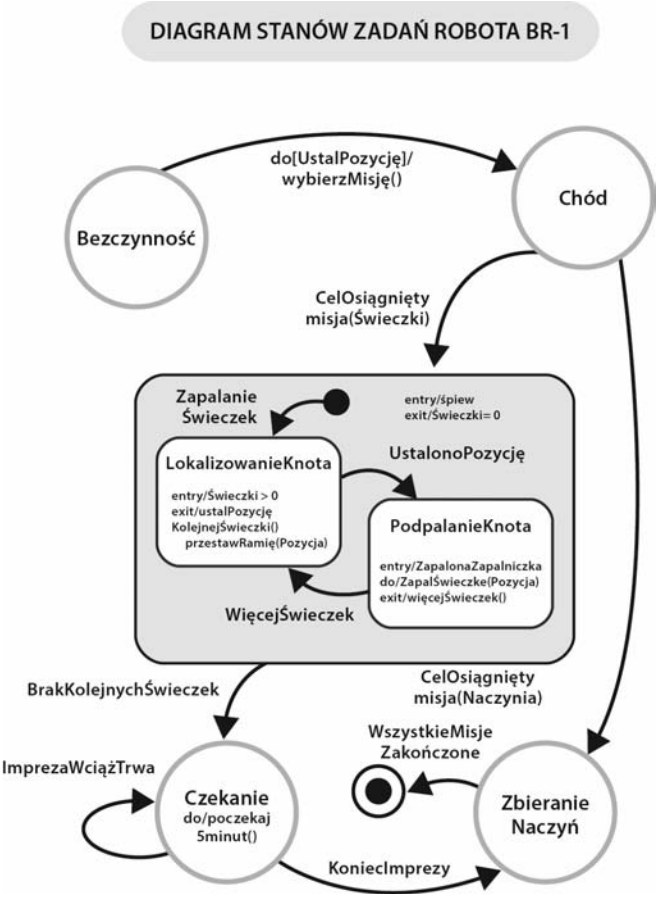
Robot pracujący w stanie chodu może dokonać dwóch przejść:

- ze stanu chodu do stanu zapalania świeczek,
- ze stanu chodu do stanu zbierania naczyń.

Stan chodu przechodzi w stan zapalania świeczek, gdy robot osiąga miejsce docelowe, a jego misja to „świeczki”. Stan chodu przechodzi w stan zbierania naczyń, gdy robot osiąga miejsce docelowe, a jego misja to „naczynia”. Aby robot wyszedł ze stanu zapalania świeczek, misja „świeczki” musi zostać zakończona. Aby wyszedł ze stanu zbierania naczyń i przeszedł do ostatniego stanu, wszystkie misje muszą zostać zakończone.

Kup książkę Poleć książkę

Rysunek 3.14.
Diagram stanów robota BR-1



`ZapalanieŚwieczek` jest stanem złożonym z dwóch podstanów: `LokalizowanieKnota` i `PodpalanieKnota`. Przed wejściem w stan zapalania świeczek określana jest wartość logiczna zmiennej `Śpiew`. Świeczki mogą zostać zapalone dopiero wówczas, gdy goście zaczną śpiewać. Operacja zapalania rozpoczyna się od zlokalizowania knota, do którego zbliża się ramię robota. Dopiero wtedy knot może zostać podpalony. W stanie `LokalizowanieKnota` akcja wejściowa wyznacza wartość logiczną wyrażenia:

- `Świeczki > 0`

Jeżeli warunek ten jest prawdziwy, to robot wychodzi z bieżącego stanu po ustaleniu pozycji kolejnej świeczki i przesuwa ramię w odpowiednie położenie (`Pozycja`).

Po ustaleniu położenia knota robot BR-1 przechodzi do stanu `PodpalanieKnota`. Przy wejściu w ten stan sprawdzane jest zapalenie zapalniczki. Jeśli zapalniczka działa, to jest ona używana do zapalenia knota świecy (stan wewnętrzny). Jeżeli `Świeczki > 0`, robot po raz kolejny wchodzi w stan `LokalizowanieKnota`. Jeżeli `Świeczki = 0`, robot przechodzi w tryb `Czekanie` i czeka na zakończenie imprezy. Po zakończeniu imprezy może zebrać wszystkie naczynia. W stanie `Czekanie` znajduje się przejście samoczynne `ImprezaWciążTrwa`. Pamiętaj, że wraz z przejściem samoczynnym wykonywane są akcje wyjściowe i wejściowe (robot wychodzi ze stanu i wchodzi w niego ponownie).

Kup książkę Poleć książkę

W tym przypadku nie ma żadnych akcji wyjściowych, a akcją wejściową jest `poczekaj5minut`. Sprawdzanym warunkiem wartowniczym jest `ImprezaWciążTrwa`. Jeżeli impreza jeszcze się nie skończyła, to robot wchodzi ponownie w ten sam stan i wykonywana jest jego akcja wejściowa. Czeka przez kolejne 5 minut. Po zakończeniu imprezy przechodzi w stan `ZbieranieNaczyń`. Jest to ostatni stan tego programu. Jeżeli wyrażenie logiczne `WszystkieMisjeZakończone` jest prawdziwe, to robot przechodzi w stan końcowy pracy. Nie wszystkie obiekty mają zdefiniowany stan końcowy — istnieją obiekty, które pracują w sposób ciągły. Diagramy stanów doskonale nadają się do definiowania zmieniających się parametrów obiektu podczas jego cyklu życiowego. Pokazują one przepływ sterowania z jednego stanu obiektu do jego innego stanu.

Co dalej?

W rozdziale 4., „Sprawdzanie rzeczywistych możliwości robota”, opiszemy zagadnienia związane z możliwościami robota. Dowiesz się, jak je poznać, a także jak sprawdzić ograniczenia mikrokontrolera, czujników, silników i efektorów końcowych.

Kup książkę Poleć książkę

Skorowidz

Kup książkę Poleć książkę

D

E

F

G

H

I

Kup książkę Poleć książkę

J

K

L

M

Kup książkę Poleć książkę

N

O

P

Kup książkę Poleć książkę

R

Kup książkę Poleć książkę

S

Ś

Kup książkę Poleć książkę

T

U

W

Z

Kup książkę Poleć książkę

Kup książkę Poleć książkę

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION

1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!
http://program-partnerski.helion.pl

GRUPA WYDAWNICZA
Helion SA

Zaprogramuj robota i stwórz maszynę przyszłości!

Roboty, te fascynujące maszyny, są coraz ważniejszym elementem naszej cywilizacji. Mają przeróżne konstrukcje i zastosowania: rozbrajają bomby, badają odległe ciała niebieskie, montują samochody i… odkurzają dywany. Wykonują prace, które dla człowieka są niebezpieczne, zbyt trudne albo po prostu męczące i nudne. Być może wkrótce w szpitalach i domach opieki zajmą się pielęgnacją obłożnie chorych. Roboty działają automatycznie, nieraz ciesząc się sporą dozą autonomii. Tym, co czyni je mniej lub bardziej inteligentnymi maszynami, jest oprogramowanie. Oznacza to, że programowanie robotów jest niezwykle cenną umiejętnością!

W tej książce przedstawiono wszystkie informacje niezbędne do rozpoczęcia samodzielnej pracy z programowaniem różnych robotów: od tych całkiem prostych aż po zaawansowane, wielofunkcyjne urządzenia. Wyjaśniono metody programowania telerobotów, robotów autonomicznych, a także strategie programowania robotów hybrydowych. Omówiono zasady programowania ruchów robota za pomocą silników i obsługi różnego rodzaju czujników. Zawarto również opis technik programowania algorytmów podejmowania decyzji przez robota, wyjaśniono też kwestie przekładania instrukcji (poleceń) z języka ludzi na język zrozumiały dla robota.

Najważniejsze zagadnienia omówione w książce:

- programowanie czujników i silników robota
- wydawanie instrukcji robotowi
- wizualny plan scenariusza pracy robota
- zaprogramowanie robota tak, aby radził sobie w niespodziewanych sytuacjach
- warunki środowiskowe, autonomia i kwestie bezpieczeństwa pracy robota
- różne techniki pracy z mikrokontrolerami LEGO Mindstorms EV3, Arduino i innymi

Cameron Hughes
programuje komputery i roboty. Obecnie zajmuje się technologiami AIM (alternatywna inteligencja maszyn) i AIR (alternatywna inteligencja robotów). Jest również programistą analitykiem na uniwersytecie stanowym Youngstown.

Tracey Hughes
jest programistką, tworzy także systemy przeznaczone do wizualizacji epistemicznej. Pracuje nad metodami graficznej wizualizacji „myślenia" robotów i komputerów.

Cameron i Tracey Hughes są członkami rady doradczej National Robotics Education Foundation.

Helion
księgarnia internetowa
http://helion.pl
zamówienia telefoniczne
0 801 339900
0 601 339900
Informatyka w najlepszym wydaniu

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
http://helion.pl

Sprawdź najnowsze promocje:
http://helion.pl/promocje
Książki najchętniej czytane:
http://helion.pl/bestsellery
Zamów informacje o nowościach:
http://helion.pl/nowosci

que

ISBN 978-83-283-2937-9

cena: 59,00 zł

sięgnij po WIĘCEJ
KOD KORZYŚCI