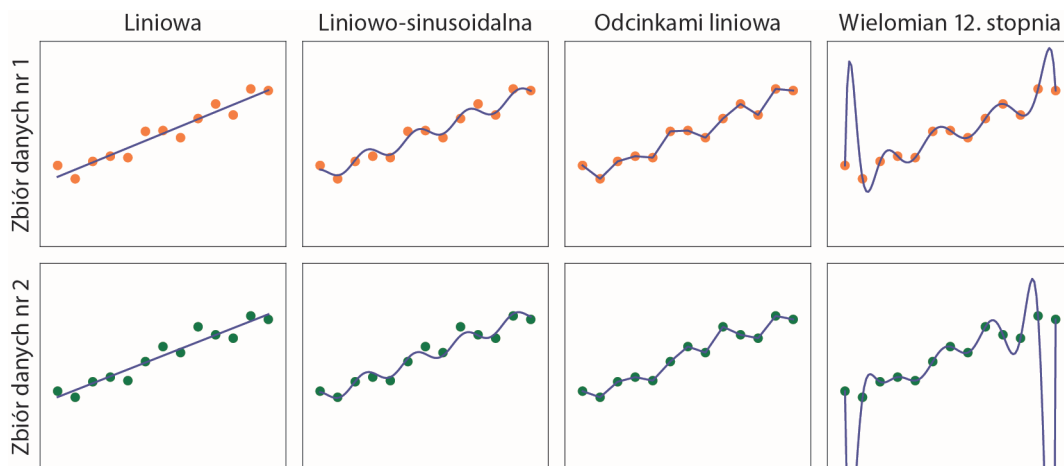
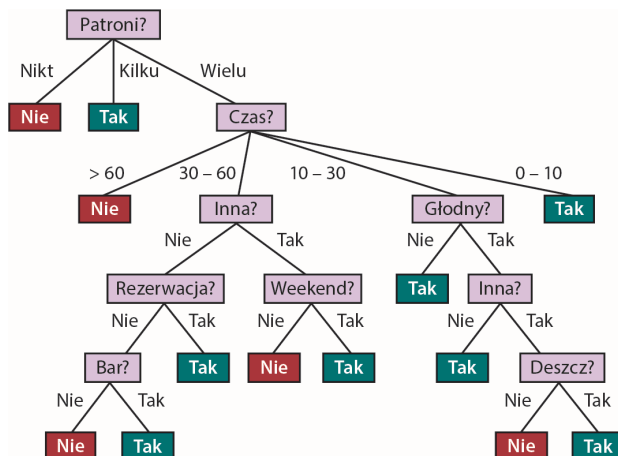


## ROZDZIAŁ 19

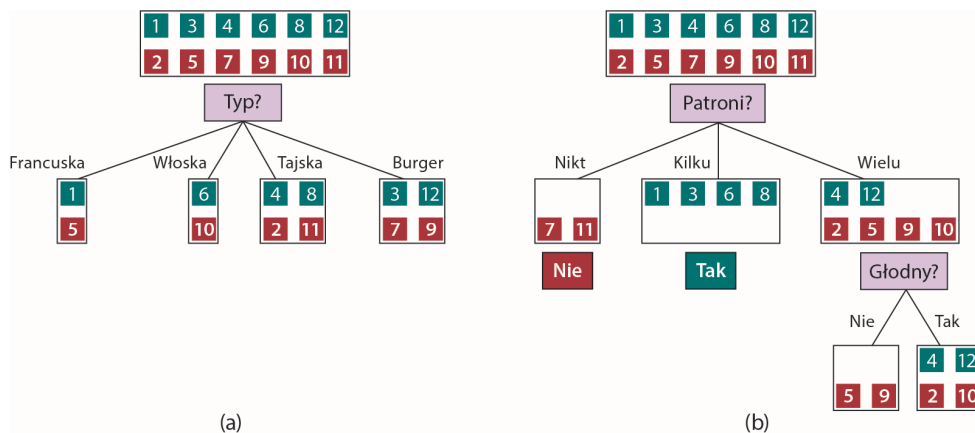
# UCZENIE MASZYNOWE Z PRZYKŁADOWYCH DANYCH



**RYSUNEK 19.1.** Znajdowanie hipotez pasujących do danych. Górny rząd: cztery wykresy najlepiej dopasowanych funkcji dla czterech hipotez wytrenowanych na tym samym zbiorze danych, pochodzących z różnych przestrzeni. Dolny rząd: te same cztery funkcje, wytrenowane na nieco odmiennym zbiorze danych (próbkowanych z tej samej funkcji  $f(x)$ )

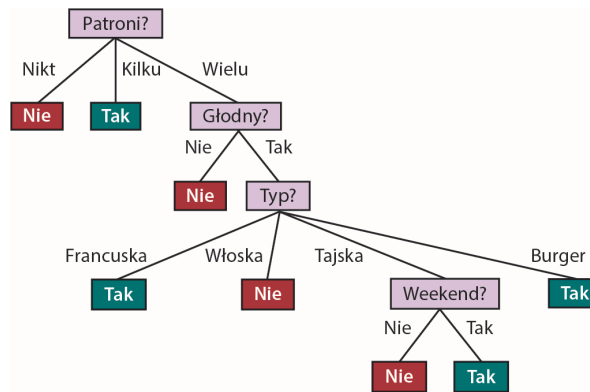


RYSUNEK 19.2. Drzewo decyzyjne dla problemu oczekiwania na stół w restauracji

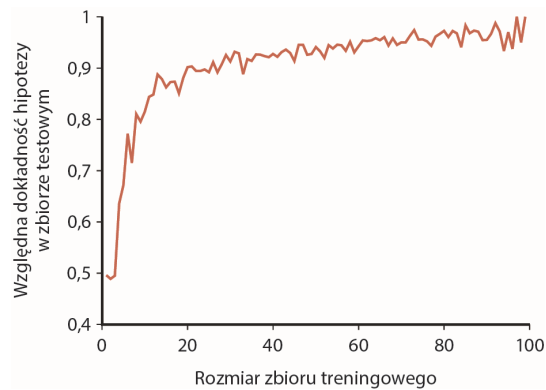


RYSUNEK 19.3. Podziały zbioru przykładów na podstawie testowania atrybutów. Dla każdego węzła pokazano pozytywne (jasne pola) i negatywne (ciemne pola) przykłady pozostałe w wyniku podziału w tym węzle. Podział według atrybutu *Typ* nie przybliża nas do rozróżniania przykładów pozytywnych od negatywnych. Podział według atrybutu *Patroni* bardzo dobrze rozdziela natomiast obie kategorie przykładów; kolejnym sugerowanym podziałem w tej sytuacji jest podział według atrybutu *Głodny*

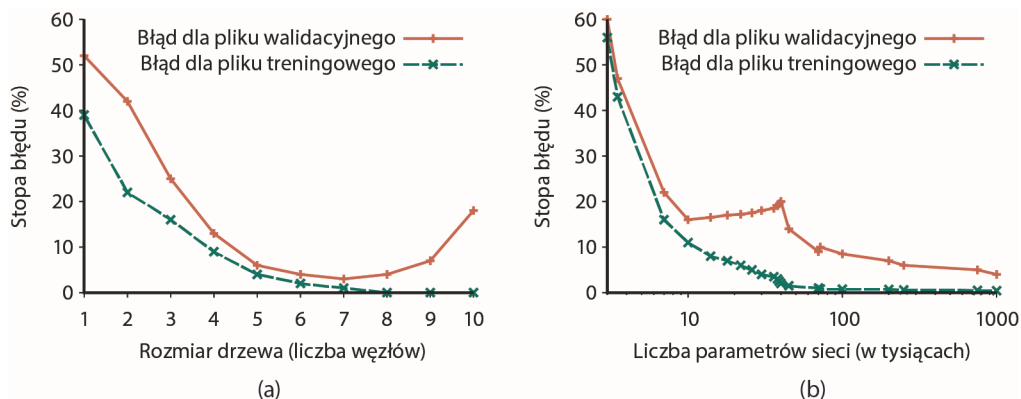




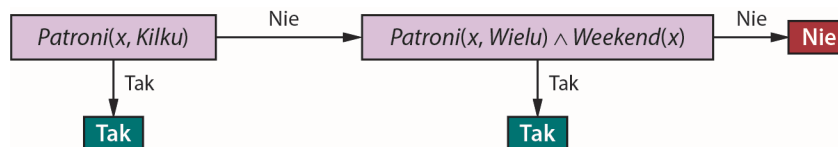
RYSUNEK 19.4. Drzewo decyzyjne indukowane przez 12-elementowy zbiór treningowy



RYSUNEK 19.5. Krzywa uczenia dla algorytmu opartego na drzewie decyzyjnym, na bazie 100 wygenerowanych losowo przykładów z problemu restauracyjnego. Każdy punkt danych jest wynikiem uśrednienia po 20 próbach

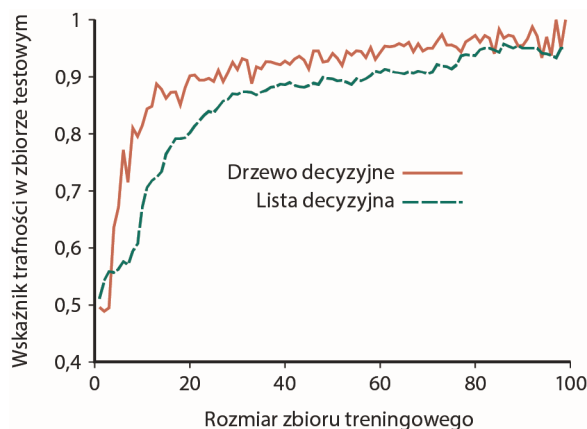


**RYSUNEK 19.6.** Stopy błędów dla zbioru treningowego (dolna, zielona linia) i zbioru walidacyjnego (górna, różowa linia) w funkcji złożoności modelu uczenia przy rozwiązywaniu dwóch różnych problemów. Algorytm MODEL-SELECTION wybiera wartość hiperparametru odpowiadającą minimum stopy błędów w zbiorze walidacyjnym. Wykres w części (a) odzwierciedla rozwiązywanie problemu restauracyjnego z sekcji 19.2.1 za pomocą drzewa decyzyjnego — optymalna wartość hiperparametru to 7 węzłów. W części (b) model uczenia jest konwolucyjną siecią neuronową (patrz podrozdział 21.3), a wartość hiperparametru jest liczbą regularnych parametrów tej sieci; rozwiązywany problem polega na rozpoznawaniu cyfr w próbkach pisma odręcznego, zebranych jako kolekcja obrazów w zbiorze danych MNIST (<http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits>)<sup>1</sup>. Optymalną wartością hiperparametru jest 1 000 000 (skala na osi poziomej jest logarytmiczna)

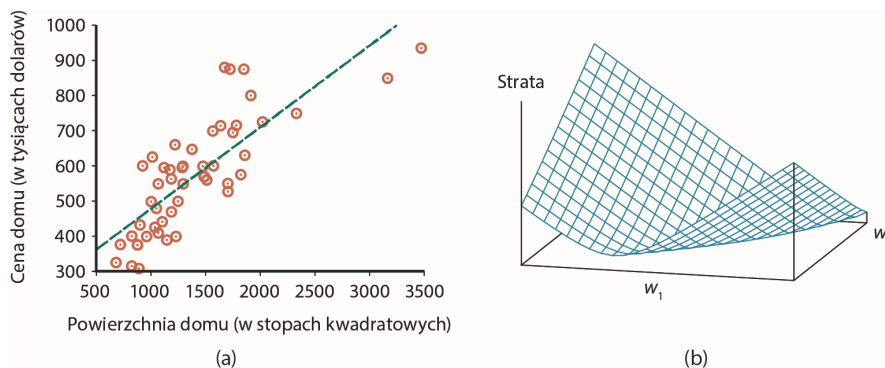


**RYSUNEK 19.7.** Lista decyzyjna dla problemu restauracyjnego

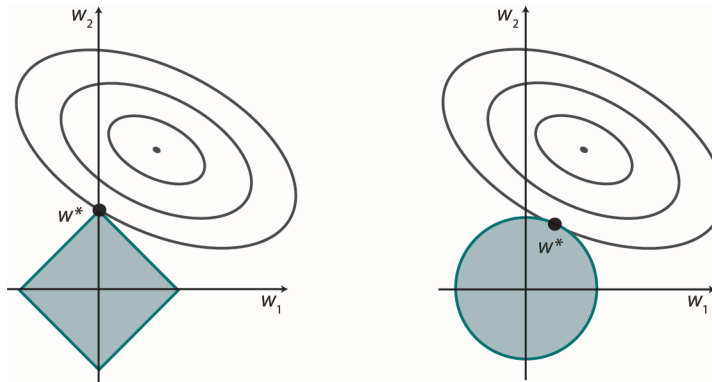
<sup>1</sup> Problem ten, a także jego rozwiązywanie przy użyciu bibliotek języka Python, opisane są szczegółowo w podrozdziale 15.6 książki *Python dla programistów. Big Data i AI. Studia przypadków*, wyd. Helion 2020, <https://helion.pl/ksiazki/pytprs.htm> — przyp. tłum.



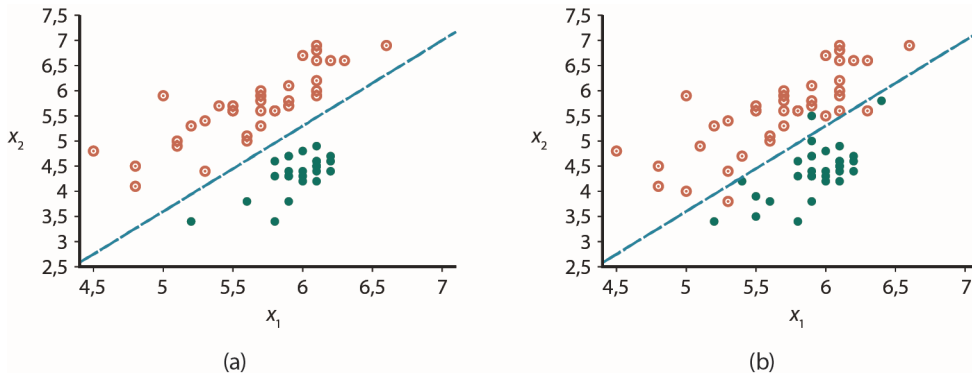
**RYСУNEK 19.8.** Krzywa uczenia dla algorytmu DECISION-LIST-LEARNING zastosowanego do problemu restauracyjnego, w porównaniu z algorytmem LEARN-DECISION-TREE. Jak widać, dla tego konkretnego problemu drzewa decyzyjne spisują się nieco lepiej



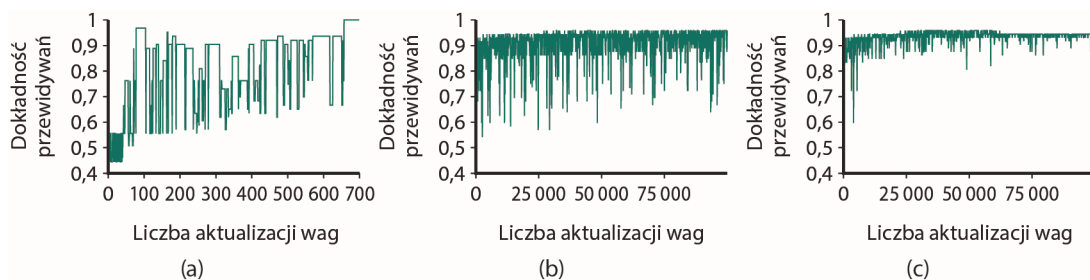
**RYСУNEK 19.9.** (a) Punkty danych odzwierciedlające cenę sprzedaży domu w funkcji jego powierzchni użytkowej (Berkeley, Kalifornia, lipiec 2009) wraz z hipotezą liniową minimalizującą funkcję straty  $L_2$ :  $y = 0,232x + 246$ . (b) Wykres funkcji straty  $\sum_j (y_j - (w_1x + w_0))^2$  dla różnych wartości  $w_0$  i  $w_1$ . Zauważmy, że funkcja straty jest wypukła — ma jedno globalne minimum



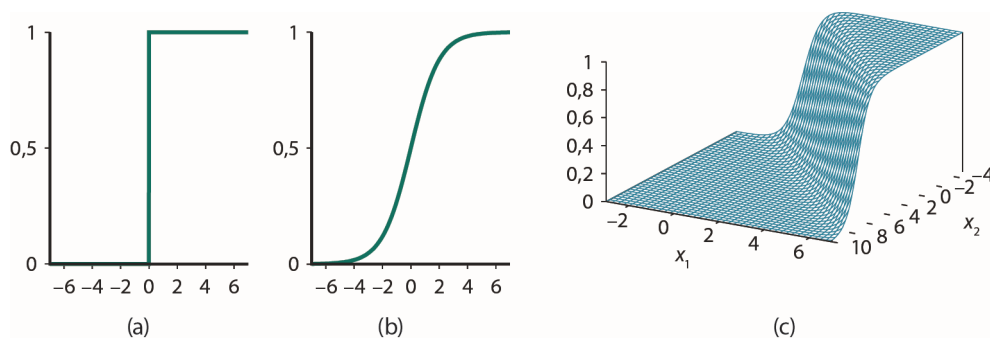
**RYSUNEK 19.10.** Poglądowe wyjaśnienie, dlaczego regularyzacja  $L_1$  przejawia tendencję do generowania modeli rzadkich. Lewa strona: w obszarze regularyzacji  $L_1$  (romb) punkt położony na jednym z koncentrycznych kręgów, reprezentujących najmniejszą osiągalną stratę, położony jest zwykle na osi, a więc ma zerową wagę. Prawa strona: obszarem regularyzacji  $L_2$  jest koło; jego punkt styczności ze wspomnianym kręgiem może znajdować się gdziekolwiek, bez związku z osiami



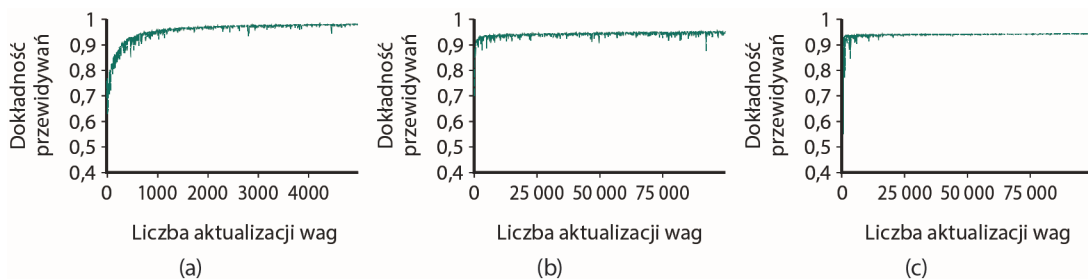
**RYSUNEK 19.11.** (a) Wykres dwóch parametrów danych sejsmicznych: magnitudy fali wglębnej (ang. *body wave*)  $x_1$  i magnitudy fali powierzchniowej (ang. *surface wave*)  $x_2$  zarejestrowanych w wyniku naturalnych trzęsień ziemi (otwarte pomarańczowe okręgi) i eksplozji nuklearnych (zielone punkty) w latach 1981 – 1990 w Azji i na Środkowym Wschodzie (Kebeasy i in., 1998). Uwidoczniono także granicę decyzyjną między obiema klasami. (b) Te same wielkości reprezentowane przez większą liczbę punktów; jak widać, dane nie są już liniowo separowalne



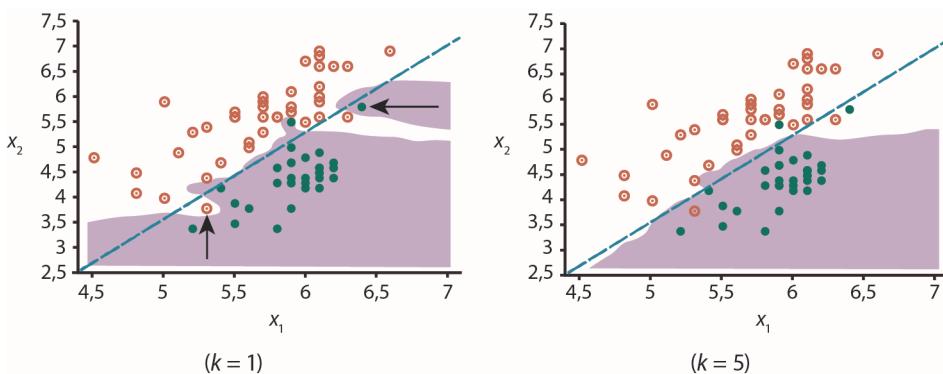
**RYСУNEK 19.12.** (a) Wykres dokładności przewidywań dla całego zbioru treningowego w funkcji liczby iteracji, w uczeniu perceptronowym, na bazie danych sejsmicznych przedstawionych w części (a) rysunku 19.11. (b) Analogiczny wykres dla zakłóconych, nieseparowalnych liniowo danych z części (b) rysunku 19.11. (c) Wykres podobny do (b), szybkością uczenia zmieniającą się w czasie według formuły  $\alpha(t) = 1000/(1000 + t)$



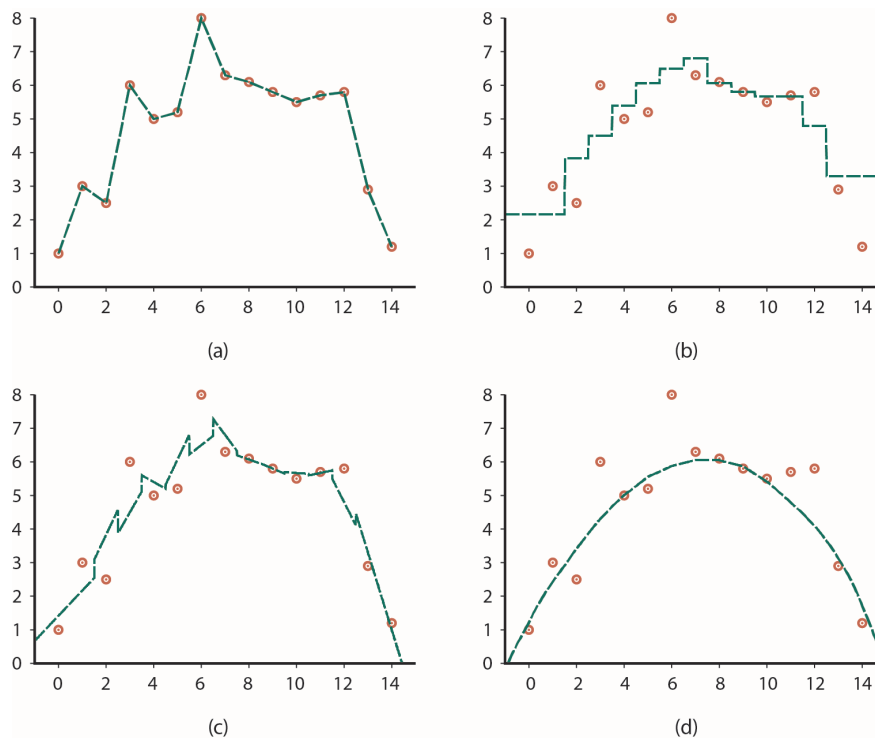
**RYСУNEK 19.13.** (a) Funkcja twardego progu  $\text{Threshold}(z)$  z wyjściem 0/1; zauważmy, że nie jest ona różniczkowalna w punkcie  $z = 0$ . (b) Funkcja logistyczna  $\text{Logistic}(z) = \frac{1}{1+e^{-z}}$ , zwana także *funkcją sigmoidalną*. (c) Wykres hipotezy stanowiącej wynik regresji logistycznej  $h_w(x) = \text{Logistic}(w \cdot x)$  dla danych z części (b) rysunku 19.11



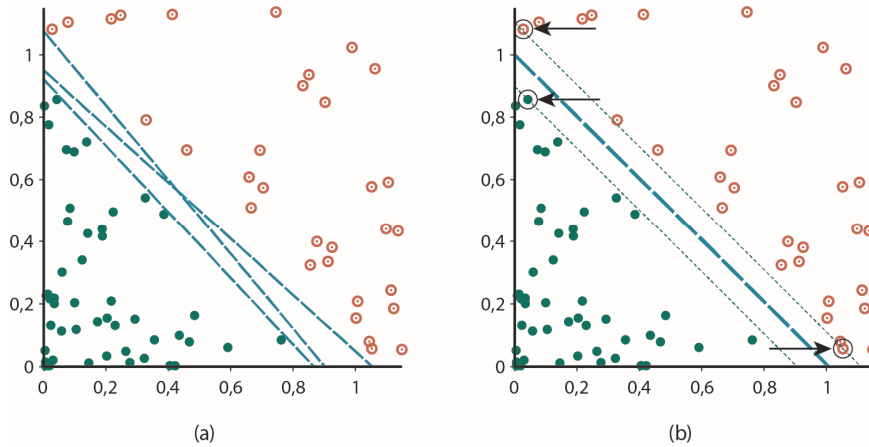
**RYSUNEK 19.14.** Powtórka eksperymentu z rysunku 19.12 z użyciem regresji logistycznej. Wykres w części (a) obejmuje 5000 iteracji (zamiast 700), liczba iteracji w częściach (b) i (c) nie zmieniła się



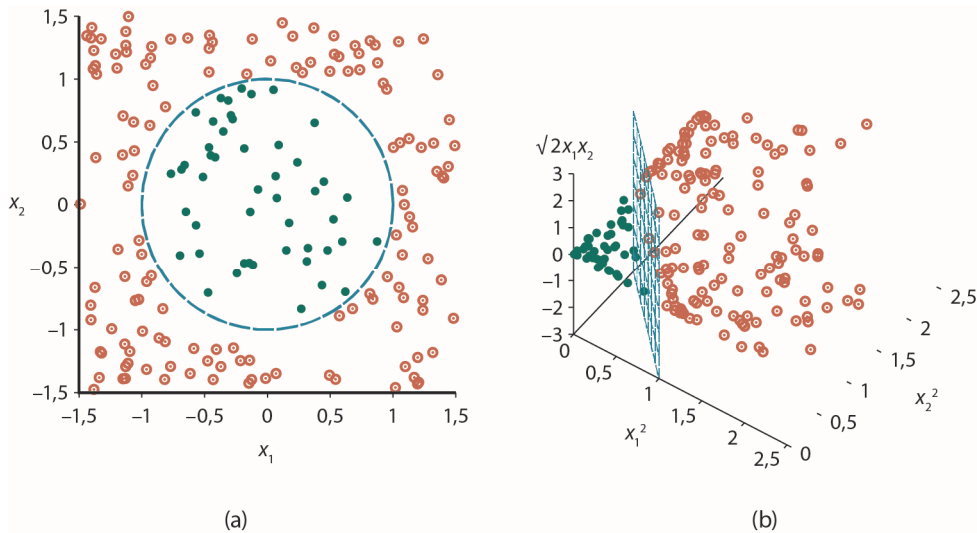
**RYSUNEK 19.15.** (a) Model  $k$  najbliższych sąsiadów ukazujący zakres klasy eksplozji nuklearnych dla danych z rysunku 19.11 dla  $k = 1$ . Przeuczenie jest ewidentne. (b) Dla  $k = 5$  problem przeuczenia znika, dla tego zbioru danych



**RYSUNEK 19.16.** Nieparametryczne modele regresyjne. (a) Połączenie punktów. (b) Uśrednienie wg 3 najbliższych sąsiadów. (c) Regresja liniowa wg 3 najbliższych sąsiadów. (d) Lokalna regresja ważona z jądrem kwadratowym o szerokości 10

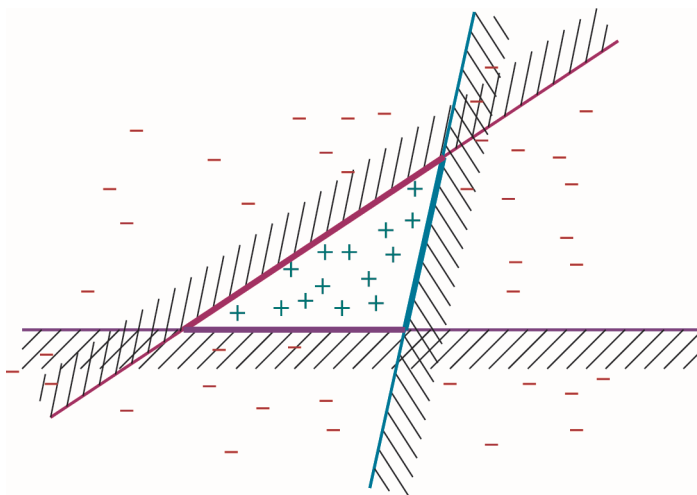


**RYSUNEK 19.17.** Klasyfikacja za pomocą maszyny SVM. (a) Dwie klasy punktów i trzy kandydatury na separatory liniowe. (b) Separator z maksymalnym marginesem (pogrubiona linia) położony jest wewnątrz marginesu (tu ograniczonego przez dwie linie przerywane). Wektory nośne (punkty otoczone większymi okręgami) to przykłady (tutaj trzy) położone najbliżej separatora.

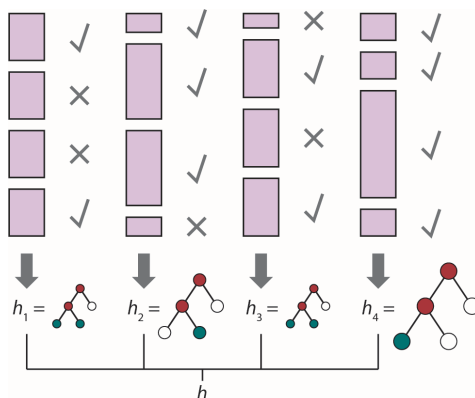


**RYSUNEK 19.18.** (a) Dwuwymiarowy zbiór treningowy z przykładami pozytywnymi (zielone punkty) i negatywnymi (pomarańczowe okręgi) wraz z rzeczywistą granicą decyzyjną (niebieskim okręgiem o równaniu  $x_1^2 + x_2^2 = 1$ ) (b) Ten sam zbiór przekształcony przez mapowanie do trójwymiarowej przestrzeni  $(x_1^2, x_2^2, \sqrt{2}x_1x_2)$ . Granica decyzyjna przekształca się do płaszczyzny położonej w przestrzeni trójwymiarowej; zbliżeniem tego widoku jest sytuacja przedstawiona w części (b) rysunku 19.17

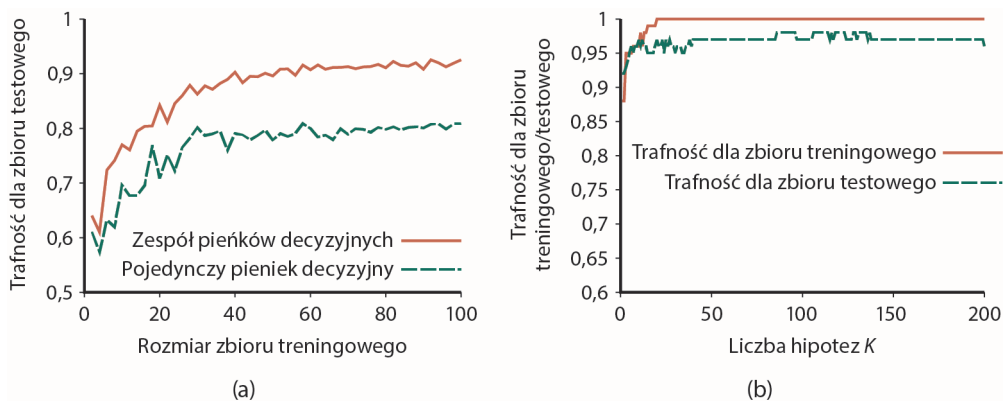




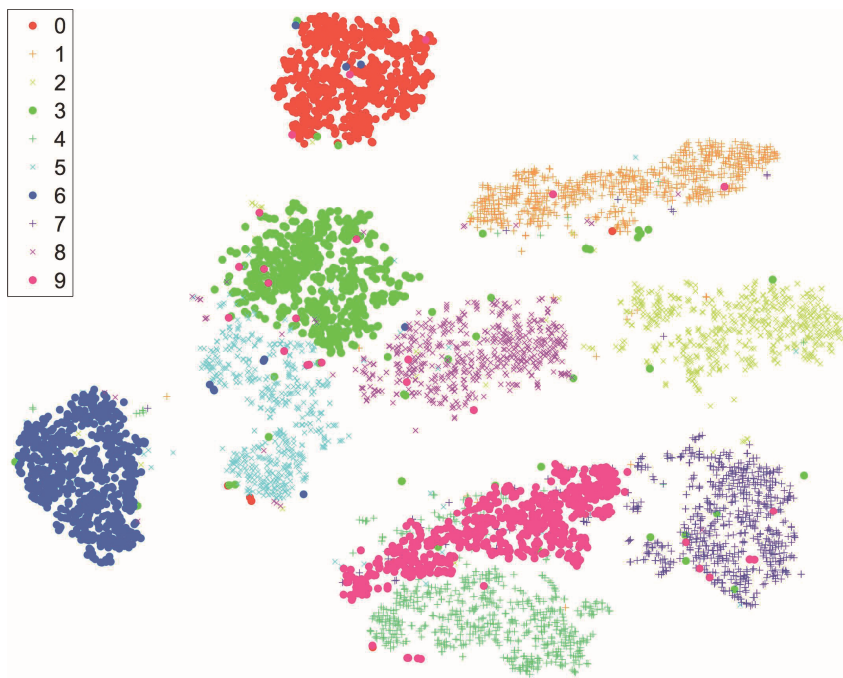
**RYSUNEK 19.19.** Ilustracja zwiększenia ekspresywności uzyskanego dzięki uczeniu zespołowemu. Widzimy kombinację trzech hipotez z progami liniowymi, każda z nich klasyfikuje przykłady pozytywne po niezakresowanej stronie progu. Agregacja tych trzech hipotez wyznacza nową hipotezę, klasyfikującą przykłady pozytywne wewnątrz trójkątnego obszaru, a przykłady negatywne na zewnątrz niego. Takiej klasyfikacji nie sposób wyrazić w kategoriach oryginalnej przestrzeni hipotez składowych



**RYSUNEK 19.20.** Tak działa *boosting*. Każdy prostokąt odpowiada jednemu przykładowi, wysokość prostokąta jest proporcjonalna do wagi tego przykładu. Symbol  $\checkmark$  oznacza prawidłowe zaklasyfikowanie przykładu przez bieżącą hipotezę, symbol  $\times$  — zaklasyfikowanie błędne. Rozmiar drzewa decyzyjnego decyduje o wadze odnośnej hipotezy w finalnym zespole



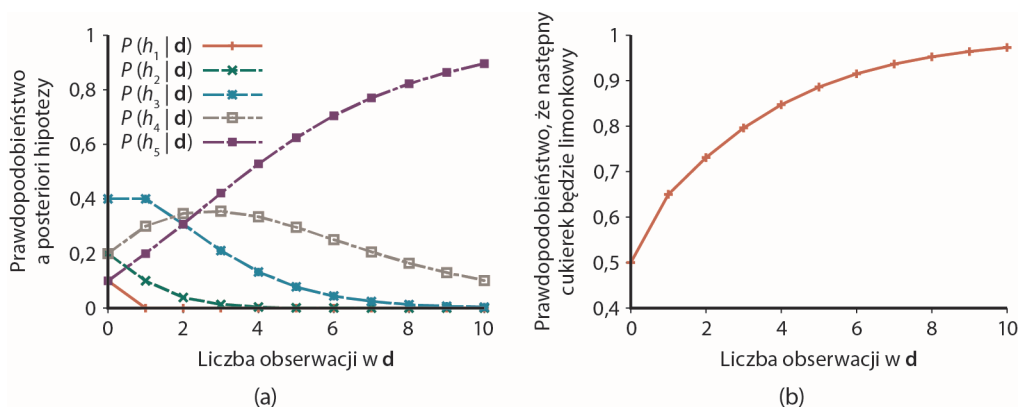
**RYSUNEK 19.21.** (a) Wykres ilustrujący wydajność boostowanego pieńka decyzyjnego dla  $K = 5$  w porównaniu z pojedynczym pieńkiem dla danych problemu restauracyjnego. (b) Trafność dla zbioru treningowego i zbioru testowego w funkcji liczby hipotez w zespole ( $K$ ). Zauważmy, że trafność dla zbioru testowego poprawia się nieznacznie nawet po tym, jak trafność dla zbioru treningowego osiągnie 100%



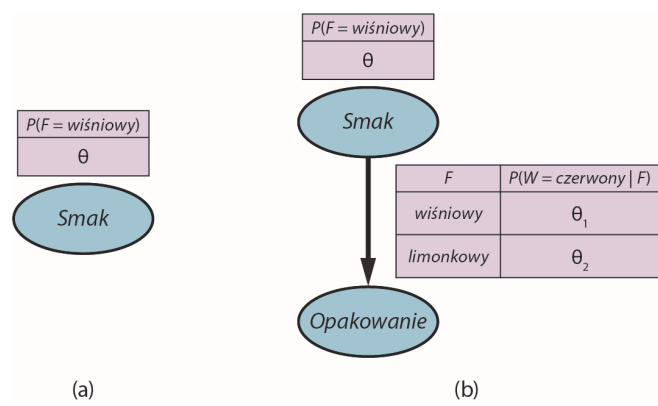
**RYSUNEK 19.22.** Dwuwymiarowa mapa t-SNE zbioru MNIST — kolekcji 60 000 obrazów przedstawiających odręcznie napisane cyfry, każdy o rozmiarze 28×28 pikseli, czyli o 784 wymiarach. Nietrudno zauważyć klasterze dla poszczególnych cyfr, z niewielkimi pomyłkami w każdym klastrze — na przykład w klastrze odpowiadającym cyfrze 0 znajduje się kilka punktów reprezentujących 3 i 6. Algorytm t-SNE znajduje reprezentację eksponującą różnice między klastrami

## ROZDZIAŁ 20

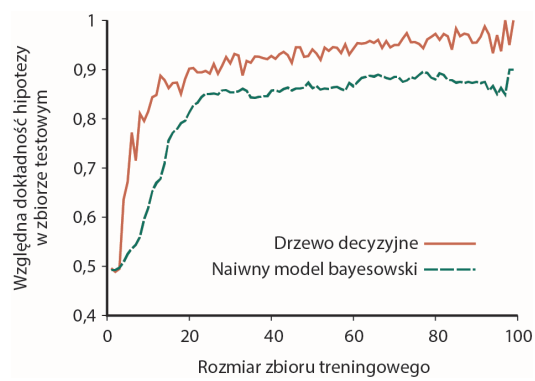
# UCZENIE MODELI PROBABILISTYCZNYCH



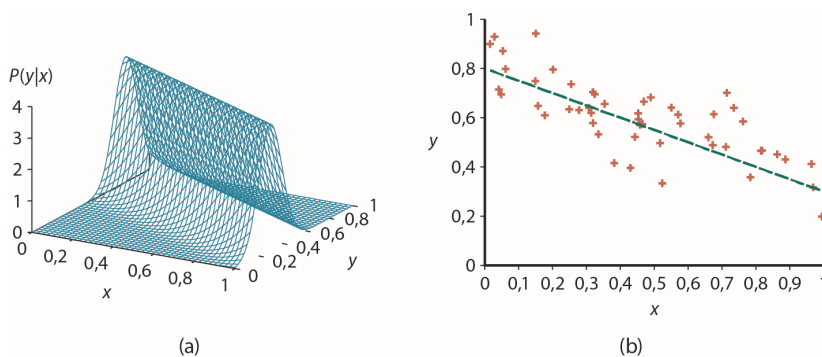
**RYСУNEK 20.1.** (a) Prawdopodobieństwa *a posteriori*  $P(h_i | d_1, \dots, d_N)$  z równania (20.1). Liczba obserwacji  $N$  zmienia się od 1 do 10, każda obserwacja dotyczy cukierka limonkowego. (b) Przewidywanie bayesowskie  $P(D_{N+1} = \text{limonkowy} | d_1, \dots, d_N)$  zgodnie z równaniem (20.2)



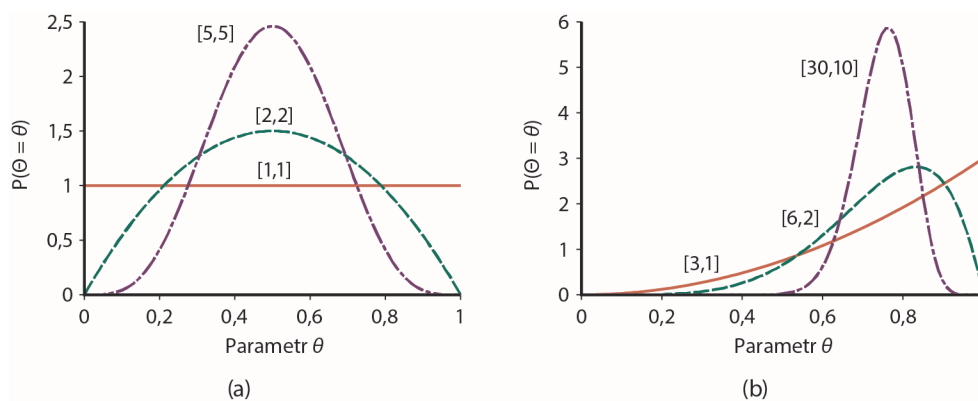
RYSUNEK 20.2. (a) Model sieci bayesowskiej dla przypadku torebki z cukierkami o nieznannej proporcji smaków. (b) Model dla przypadku, gdy kolor opakowania powiązany jest (probabilistycznie) ze smakiem cukierka



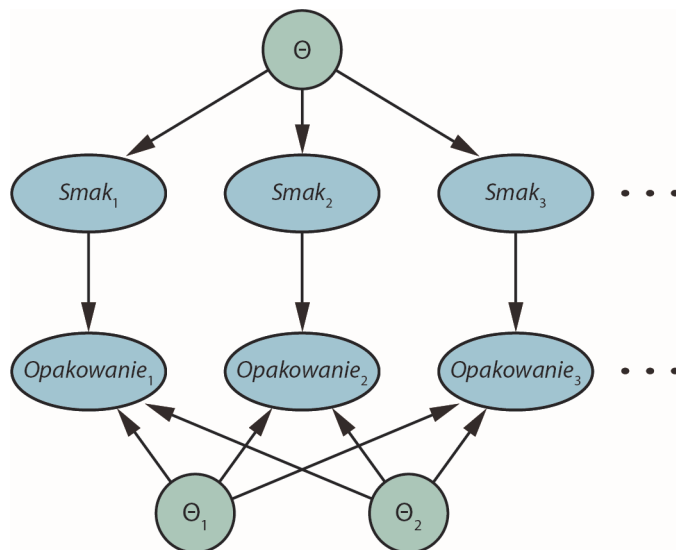
RYSUNEK 20.3. Krzywa uczenia naiwnego uczenia bayesowskiego dla problemu restauracyjnego z rozdziału 19. Dla porównania — krzywa uczenia przy użyciu drzew decyzyjnych dla tego samego problemu



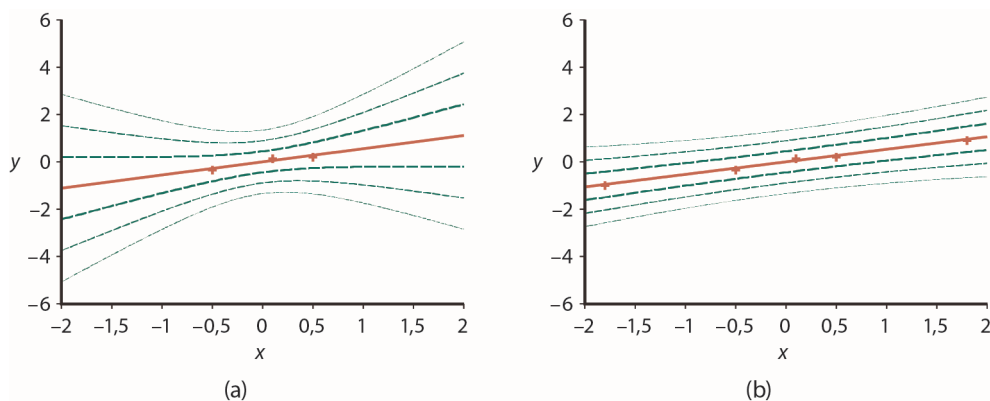
**RYSUNEK 20.4.** (a) Liniowy model gaussowski opisywany przez równanie  $y = \theta_1 x + \theta_2$  z szumem gaussowskim o ustalonej wariancji. (b) Zbiór 50 punktów danych wygenerowanych przez ten model i linia najlepszego dopasowania



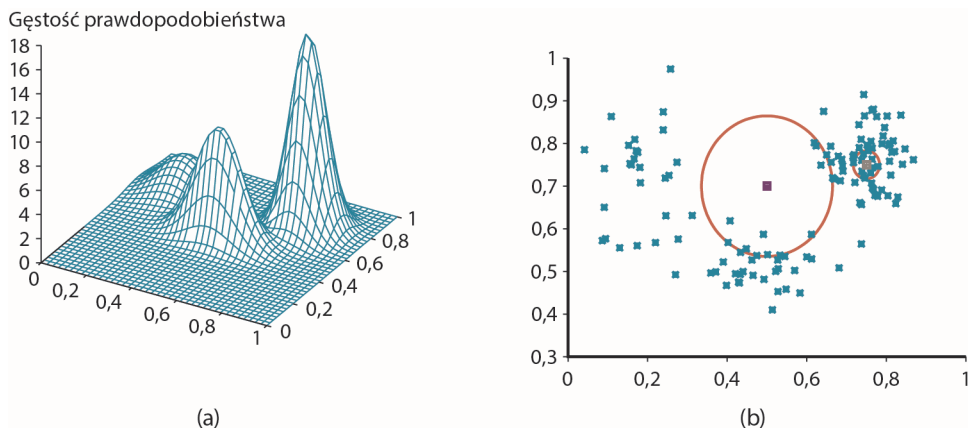
**RYSUNEK 20.5.** Przykłady rozkładów  $Beta(a, b)$  dla różnych wartości  $a$  i  $b$



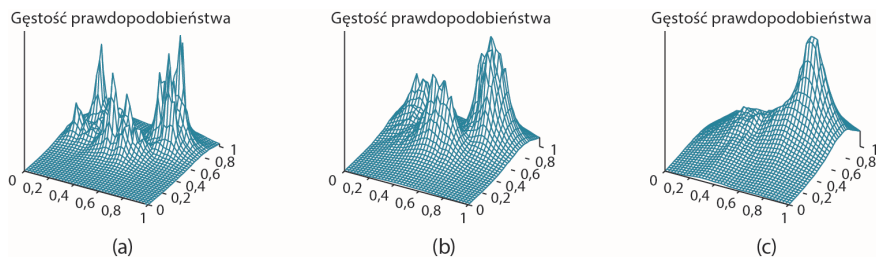
**RYSUNEK 20.6.** Sieć bayesowska odpowiadająca procesowi uczenia bayesowskiego. Prawdopodobieństwa *a posteriori* zmiennych parametrycznych  $\Theta$ ,  $\Theta_1$  i  $\Theta_2$  można wywnioskować z ich rozkładów *a priori* oraz zmiennych dowodowych  $Smak_i$  i  $Opakowanie_i$



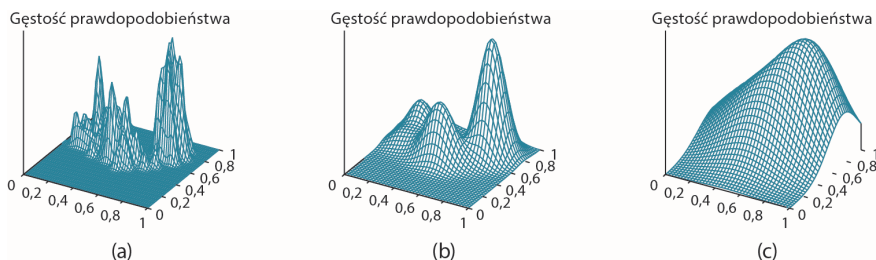
**RYSUNEK 20.7.** Bayesowska regresja liniowa z modelem ograniczonym do wykresu przechodzącego przez początek układu współrzędnych i z ustaloną wariancją szumu  $\sigma^2 = 0,2$ . Pokazano kontury dla  $\pm 1$ ,  $\pm 2$  i  $\pm 3$  odchyłeń standardowych dla przewidywanej gęstości. (a) Przy trzech punktach danych w pobliżu punktu  $(0, 0)$  nachylenie jest dość niepewne, przy  $\sigma_N^2 \approx 0,3861$ . Zwróćmy uwagę na to, jak niepewność ta wzrasta wraz z odległością od obserwowanych punktów danych. (b) Dodanie dwóch odległych punktów skutkuje ściślejszym ograniczeniem nachylenia  $\theta$  przy  $\sigma_N^2 \approx 0,0286$ . Pozostała wariancja przewidywanej gęstości jest niemal całkowicie wyznaczona przez stały szum  $\sigma^2 = 0,2$



**RYSUNEK 20.8.** (a) Wykres 3D mieszanki gaussowskiej z części (a) rysunku 20.12. (b) Próbkę 128 punktów z mieszanki wraz z dwoma badanymi punktami (małe kwadraty) i 10 najbliższymi sąsiadami każdego z nich (duży i mały okrąg)

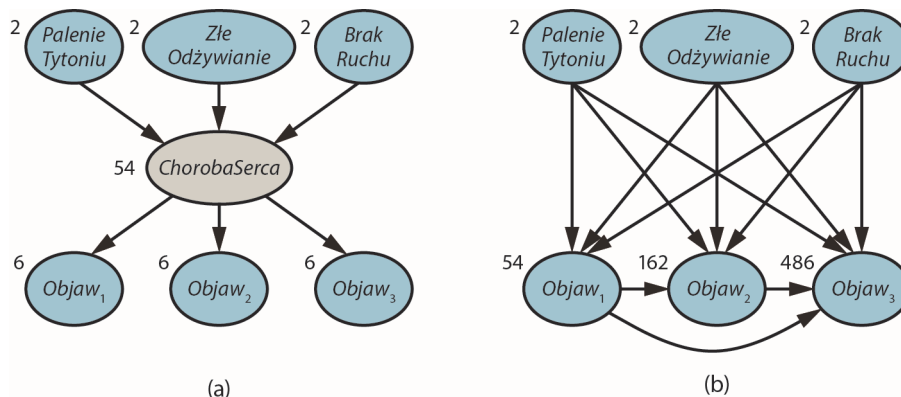


**RYSUNEK 20.9.** Estymowanie gęstości za pomocą  $k$  najbliższych sąsiadów w zastosowaniu do danych z części (b) rysunku 20.8, dla  $k = 3, 10$  i  $40$ . Dla  $k = 3$  wykres jest zbyt szpiczasty, dla  $k = 40$  jest zbyt gładki,  $k = 10$  wydaje się bardzo dobrym przybliżeniem. Do znalezienia najlepszej wartości  $k$  można wykorzystać walidację krzyżową

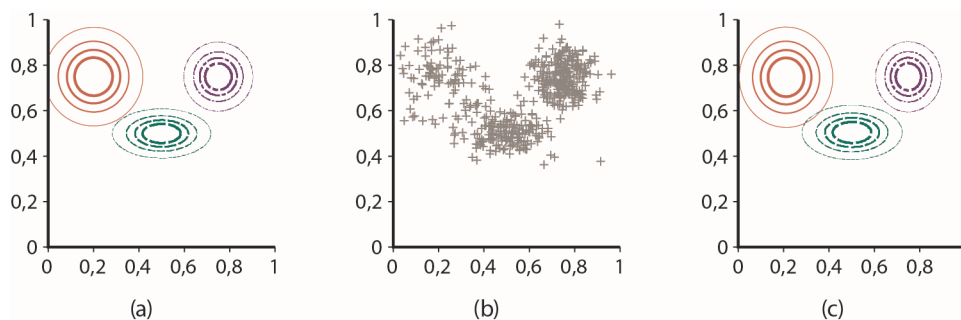


**RYSUNEK 20.10.** Estymowanie funkcji gęstości prawdopodobieństwa przy użyciu funkcji jądra dla danych z części (b) rysunku 20.8, z wykorzystaniem jąder gaussowskich dla  $w = 0,02, 0,07$  i  $0,20$ ;  $w = 0,07$  wydaje się najlepszym wyborem

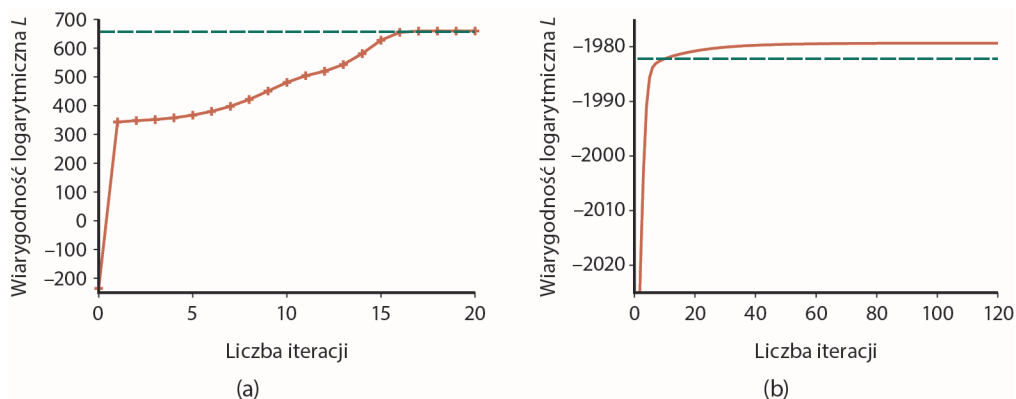




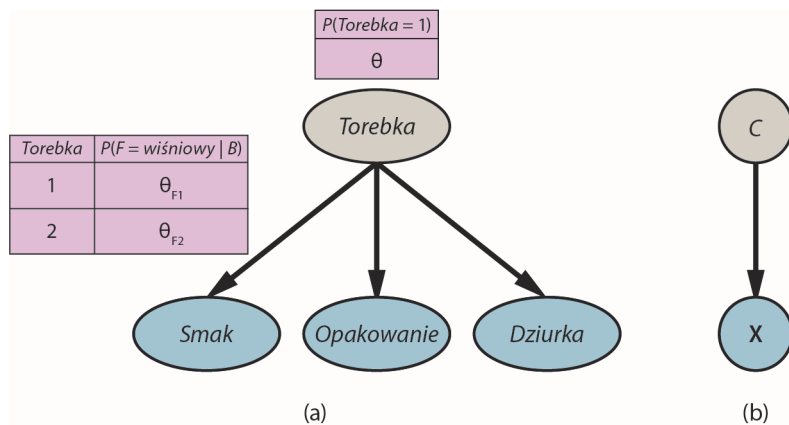
**RYSUNEK 20.11.** (a) Prosta sieć diagnostyczna dla rozpoznawania choroby serca, reprezentowanej przez zmienną ukrytą. Każda zmienna może przyjmować trzy wartości i etykietowana jest przez liczbę niezależnych parametrów w jej rozkładzie warunkowym; łączna liczba parametrów wynosi 78. (b) Równoważna sieć po usunięciu zmiennej *ChorobaSerca*. Zauważmy, że zmienne reprezentujące objawy przestały być warunkowo niezależne wobec swych rodziców. Ta sieć wymaga 708 parametrów



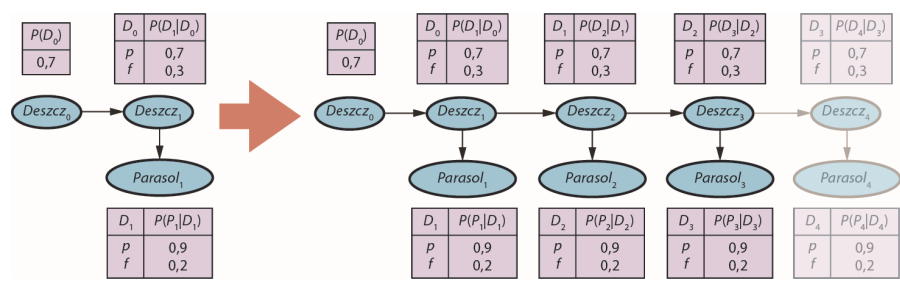
**RYSUNEK 20.12.** (a) Model mieszanek gaussowskich z trzema komponentami, posiadającymi wagi (odpowiednio od lewej do prawej) 0,2, 0,3 i 0,5. (b) Zbiór 500 punktów danych próbkowanych z rozkładu w części (a). (c) Model zrekonstruowany na podstawie danych z części (b)



**RYSUNEK 20.13.** Wykresy przedstawiające wiarygodność logarytmiczną danych ( $L$ ) jako funkcję liczby iteracji wykonanych przez algorytm EM; linia pozioma przedstawia wiarygodność logarytmiczną zgodnie z prawdziwym modelem. (a) Wykres dla modelu mieszanki gaussowskiej z rysunku 20.12. (b) Wykres dla sieci bayesowskiej z części (a) rysunku 20.14



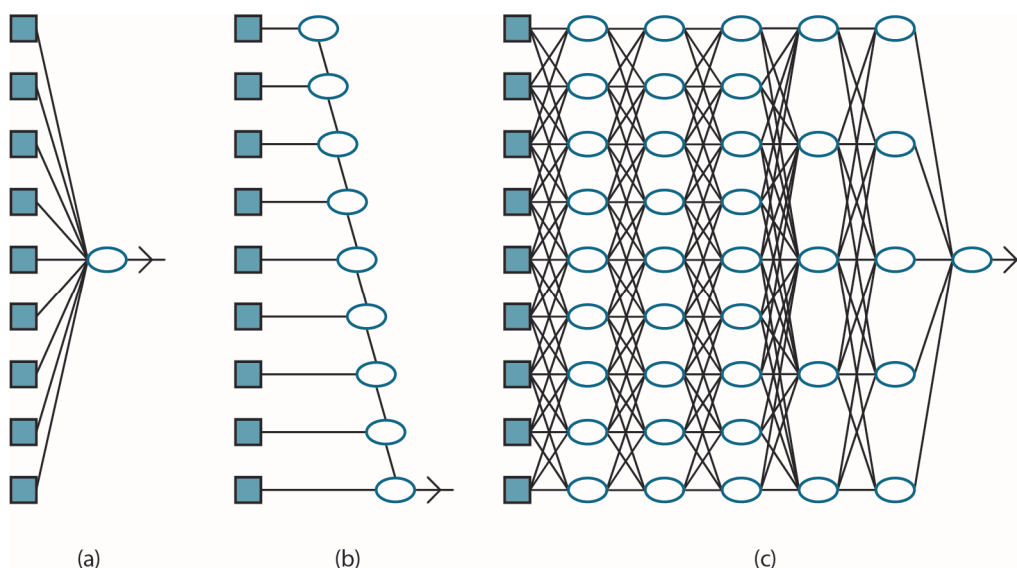
**RYSUNEK 20.14.** (a) Model mieszanki dla historyjki z cukierkami. Proporcje różnych smaków, kolorów opakowań i obecności/braku dziurki nie są obserwowane, mogą być różne dla różnych torebek. (b) Sieć bayesowska dla mieszanki gaussowskiej. Średnia i kowariancja obserwowanych zmiennych  $\mathbf{X}$  zależne są od komponentu  $C$



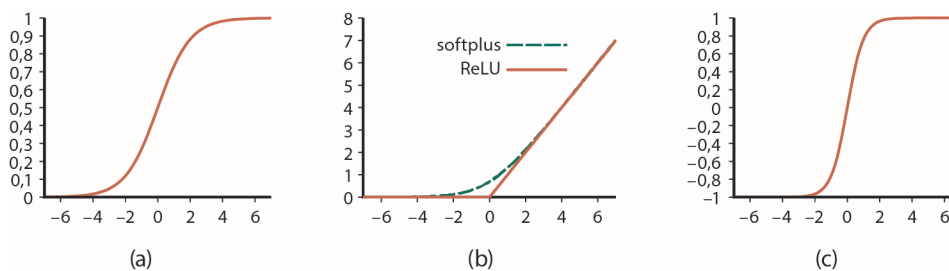
**RYСУNEK 20.15.** Rozwinięta dynamiczna sieć bayesowska reprezentująca ukryty łańcuch Markowa (reprodukcja rysunku 14.14)

## ROZDZIAŁ 21

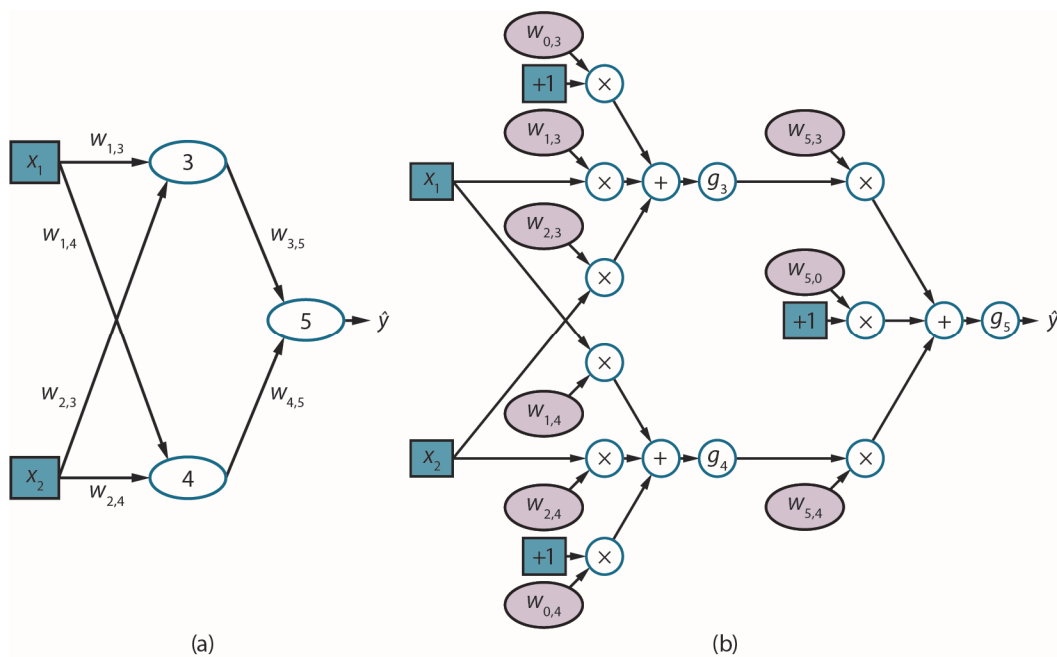
# GŁĘBOKIE UCZENIE



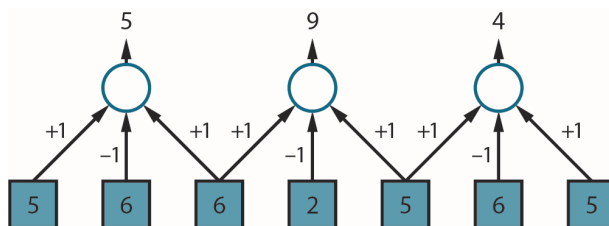
**RYСУNEK 21.1.** (a) Płytki model, taki jak regresja liniowa, charakteryzuje się krótkimi ścieżkami obliczeniowymi między wejściami a wyjściami. (b) W sieci list decyzyjnych (patrz podrozdział 19.5) ścieżki obliczeniowe mogą być dłuższe dla niektórych wejść, jednak większość ścieżek to krótkie ścieżki. (c) W sieci głębokiego uczenia ścieżki generalnie są długie, dzięki czemu każda zmienna może oddziaływać z pozostałymi



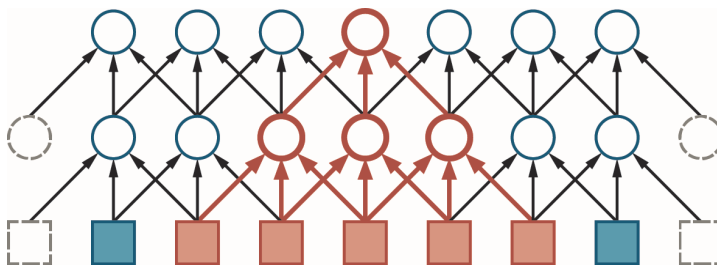
**RYSUNEK 21.2.** Funkcje aktywacyjne najczęściej wykorzystywane w systemach głębokiego uczenia (a) funkcja logistyczna (sigmoidalna) (b) funkcje ReLU i softplus (c) funkcja tanh



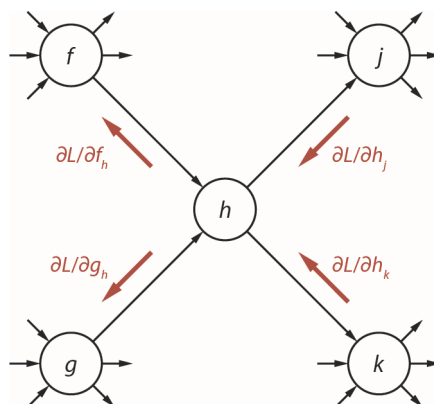
**RYSUNEK 21.3.** (a) Sieć neuronowa z dwoma wejściami, jedną warstwą ukrytą zawierającą dwie jednostki i jedną jednostką wyjściową. Nie pokazano fikcyjnych wejść i związanych z nimi wag. (b) Ta sama sieć rozpakowana do pełnego grafu obliczeniowego



**RYSUNEK 21.4.** Przykład jednowymiarowej operacji konwolucji z jądrem o rozmiarze  $l = 3$  i krokiem  $s = 2$ . Maksymalna odpowiedź wycelowana jest na najciemniejszym (najmniej intensywnym) pikselu wejściowym. Wynik przekazywany jest zwykle do nieliniowej funkcji aktywacyjnej (nie pokazano jej na rysunku) przed dostarczeniem do następnej ukrytej warstwy



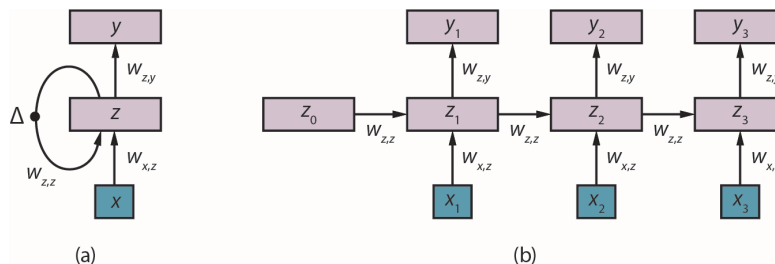
**RYSUNEK 21.5.** Dwie pierwsze warstwy sieci CNN dla jednowymiarowego obrazka z jądrem o rozmiarze  $l = 3$  i krokiem  $s = 2$ . Na obu końcach dodano dopełnienie w celu utrzymania rozmiaru warstw ukrytych identycznego z rozmiarem warstwy wejściowej. Kolorem czerwonym wyróżniono pole recepcyjne jednostki w drugiej ukrytej warstwie. Ogólnie rzecz biorąc, im głębsza jednostka, tym większe jej pole recepcyjne



**RYSUNEK 21.6.** Ilustracja propagacji wstecznej informacji o gradiencie w pewnym grafie obliczeniowym. Obliczanie wyjścia sieci przebiega od strony lewej do prawej, propagacja wsteczna gradientów przebiega od prawej do lewej



**RYSUNEK 21.7.** Błąd dla zbioru testowego w funkcji szerokości warstwy (mierzonej liczbą wag), dla sieci konwolucyjnej o trzech i o jedenastu warstwach. Dane zaczerpnięte z wczesnych wersji systemu transkrypcji adresów Google Street View (Goodfellow i in., 2014)



**RYSUNEK 21.8.** (a) Schematyczny diagram podstawowej sieci RNN, w której ukryta warstwa  $z$  zawiera rekurencyjne połączenia; symbol  $\Delta$  reprezentuje opóźnienie. (b) Ta sama sieć rozwinięta na trzy kroki czasowe, do postaci sieci ze sprzężeniem w przód. Zauważmy, że wagi współdzielone są przez wszystkie kroki czasowe

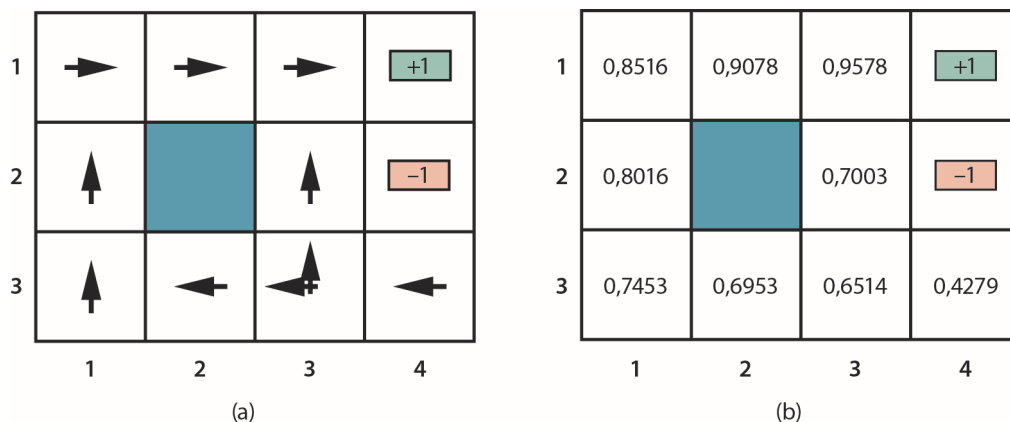


**RYSUNEK 21.9.** Demonstracja możliwości modelu generatywnego. Model wytrenowany został do poruszania się w różnych kierunkach przestrzeni  $z$  reprezentujących różne aspekty twarzy, a przedstawione obrazy, wszystkie wygenerowane przez ów model, ilustrują, co będzie się działo, gdy będziemy dekodować różne punkty tej przestrzeni i wykonywać na nich operacje arytmetyczne. Rozpoczynamy od współrzędnych odpowiadających koncepcji „mężczyzna w okularach”, odejmujemy od nich współrzędne odpowiadające koncepcji „mężczyzna” i dodajemy współrzędne odpowiadające koncepcji „kobieta”, otrzymując w efekcie współrzędne dla koncepcji „kobieta w okularach”. Obrazy zaczerpnięto z artykułu (Radford i in., 2015) za zgodą autorów

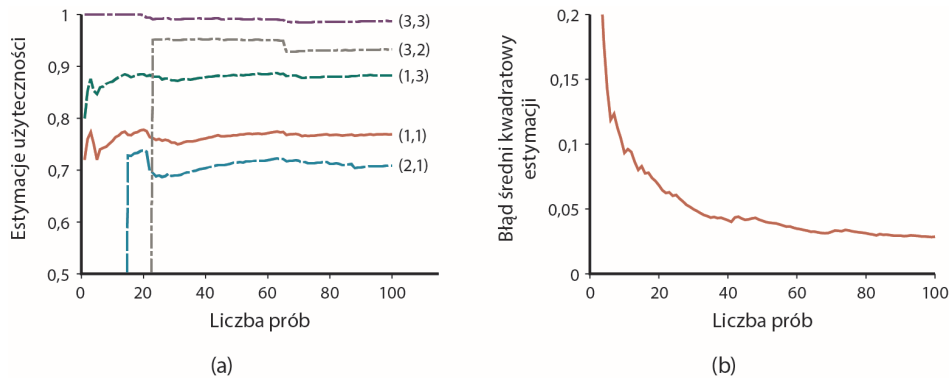


## ROZDZIAŁ 22

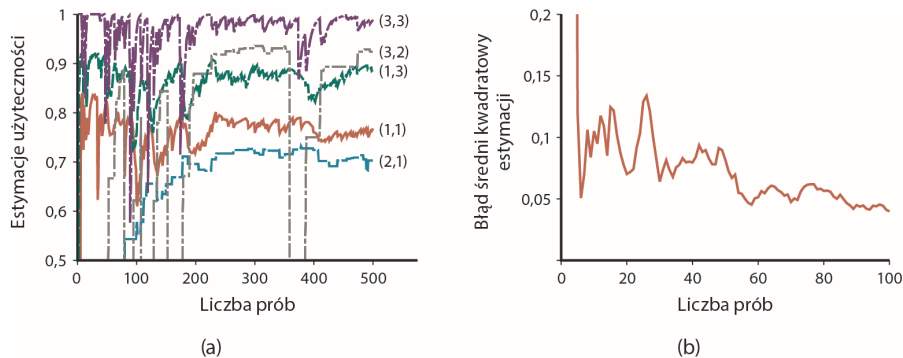
# UCZENIE ZE WZMACNIANIEM



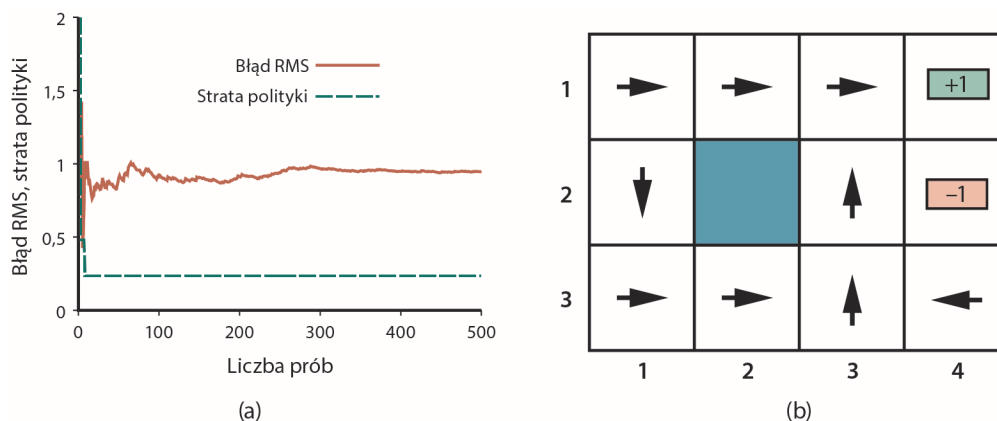
**RYSUNEK 22.1.** (a) Optymalne polityki dla środowiska stochastycznego z funkcją nagrody  $R(s, a, s') = -0,04$  dla przejść między stanami niekończącymi. Optymalne polityki są dwie, ponieważ w stanie (3, 1) optymalne są oba ruchy w *lewo* i w *górę*; widzieliśmy to już wcześniej, na rysunku 17.2. (b) Użyteczności poszczególnych stanów przy założeniu danej polityki  $\pi$



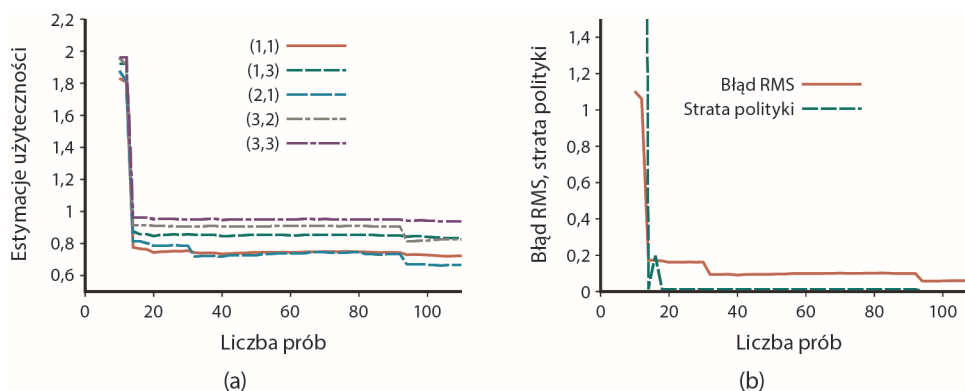
**RYSUNEK 22.3.** Krzywe uczenia pasywnego agenta ADP dla środowiska z rysunku 22.1 przy założeniu optymalnej polityki przedstawionej na tym rysunku. (a) Estymacje użyteczności dla wybranego podzbioru stanów jako funkcja liczby prób. Zauważmy, że wymagane jest wykonanie (odpowiednio) 14 i 23 prób, by rzadko odwiedzane stany (2, 1) i (3, 2) „odkryły”, że połączone są ze stanem końcowym (4, 3) o nagrodzie +1. (b) Średni błąd kwadratowy estymacji  $U(1, 1)$  uśrednionej po 50 przebiegach, z których każdy obejmuje 100 prób



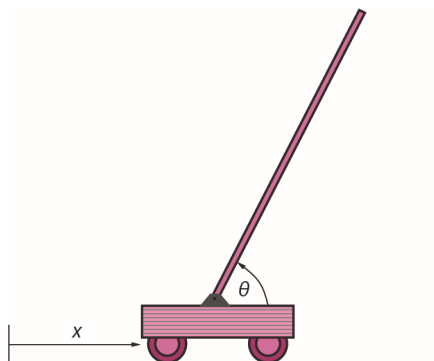
**RYSUNEK 22.4.** Krzywe uczenia TD dla środowiska z rysunku 22.1. (a) Estymacje użyteczności wybranego podzbioru stanów, jako funkcja liczby prób dla pojedynczego przebiegu obejmującego 500 prób; porównaj to z pojedynczym przebiegiem obejmującym 100 prób w części (a) rysunku 22.3. (b) Średni błąd kwadratowy estymacji  $U(1, 1)$  uśrednionej po 50 przebiegach, z których każdy obejmuje 100 prób



**RYSUNEK 22.5.** Wydajność zachłannego agenta ADP wykonującego akcję rekomendowaną przez optymalną politykę dla wyuczonego modelu. (a) Średni błąd kwadratowy (RMS) uśredniony po wszystkich dziewięciu stanach niekończących i strata polityki w stanie (1, 1). Widzimy szybką zbieżność polityki — po zaledwie ośmiu próbach — do suboptymalnej polityki o stracie 0,235. (b) Suboptymalna polityka, do której zmierza zachłanny agent w tej konkretnej sekwencji prób. Zwróćmy uwagę na akcję w *dół* w stanie (1, 2)



**RYSUNEK 22.6.** Wydajność eksploracyjnego agenta ADP wobec  $R^+ = 2$  i  $N_e = 5$ . (a) Zmiana w czasie estymacji użyteczności dla wybranych stanów. (b) Błąd RMS wartości użyteczności i powiązana strata polityki



(a)



(b)

**RYSUNEK 22.7.** Konfiguracja dla problemu balansującego długiego drążka osadzonego na ruchomym wózku. Wózek może być przesuwany w lewo i w prawo przez kontroler obserwujący położenie wózka ( $x$ ) i jego prędkość ( $\dot{x}$ ) oraz nachylenie drążka w stosunku do poziomu ( $\theta$ ) i szybkość zmiany tego nachylenia ( $\dot{\theta}$ ). (b) Sześć nałożonych na siebie zdjęć poklatkowych pojedynczego autonomicznego helikoptera wykonującego bardzo trudny manewr krążenia wokół celu (*nose-in-circles*)<sup>2</sup>. Helikopter sterowany jest zgodnie z polityką ustaloną przez algorytm wyszukiwania polityk PEGASUS (Ng i in., 2003). Na podstawie obserwacji zachowania się prawdziwego helikoptera wskutek różnych manipulacji jego przyrządami opracowano model symulatora i uruchomiono na nim wspomniany algorytm na wiele godzin. Opracowano różne kontrolery do różnych manewrów, we wszystkich przypadkach wydajność kontrolera znacznie przewyższała wydajność doświadczonego człowieka-pilota manewrującego zdalnie helikopterem. (Zdjęcie dzięki uprzejmości Andrew Ng)

<sup>2</sup> Patrz <https://www.youtube.com/watch?v=hUpwbGEUyM0> — przyp. tłum.

## ROZDZIAŁ 23

# PRZETWARZANIE JĘZYKA NATURALNEGO

### WordNet Search - 3.1

- [WordNet home page](#) - [Glossary](#) - [Help](#)

Word to search for:

Display Options:

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations  
Display options for sense: {frequency} {offset} <lexical filename > [lexical file number]  
(gloss) "an example sentence"  
Display options for word: word#sense number (sense key)

**Noun**

- (7){02125600} <noun.animal>[05] [S: \(n\)](#) **kitten#1** (**kitten%1:05:00::**), [kitty#3](#) ([kitty%1:05:02::](#)) (young domestic cat)

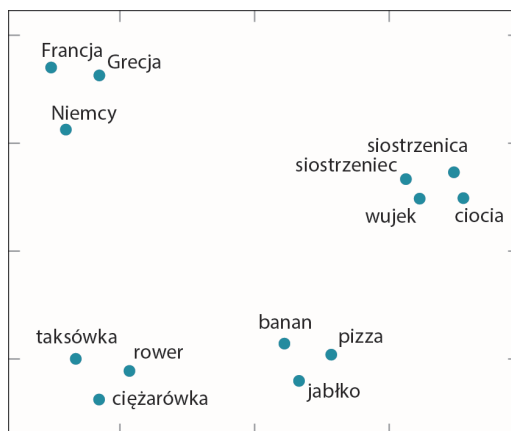
**Verb**

- {00057849} <verb.body>[29] [S: \(v\)](#) **kitten#1** (**kitten%2:29:00::**) (give birth to kittens)  
"our cat *kittened* again this year"

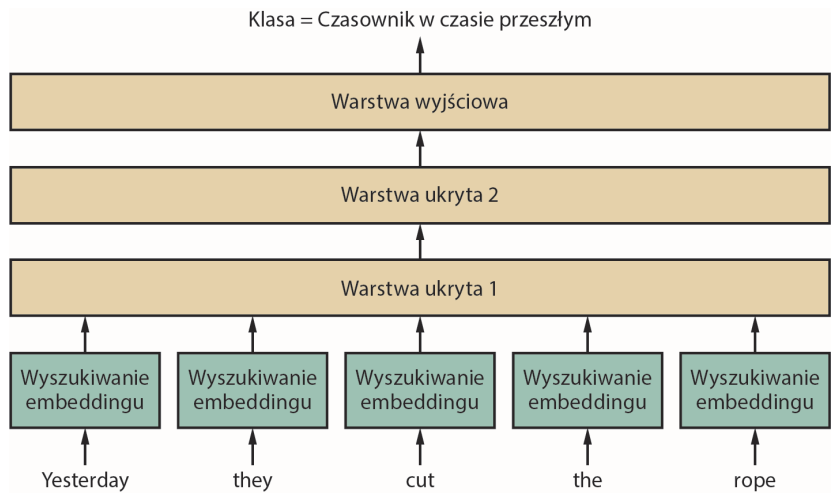
RYSUNEK 23.1. Wynik przykładowego wyszukiwania w słowniku WordNet

## ROZDZIAŁ 24

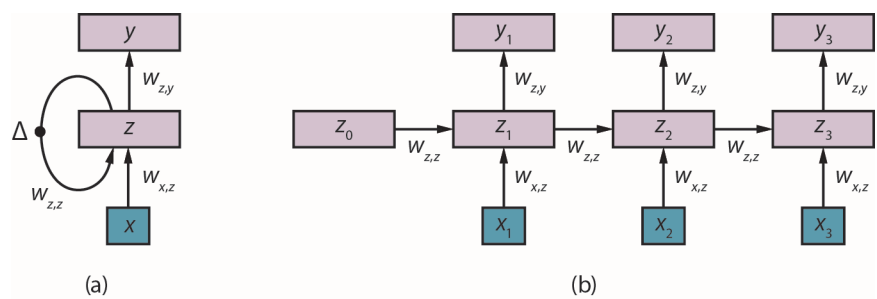
# GŁĘBOKIE UCZENIE W PRZETWARZANIU JĘZYKA NATURALNEGO



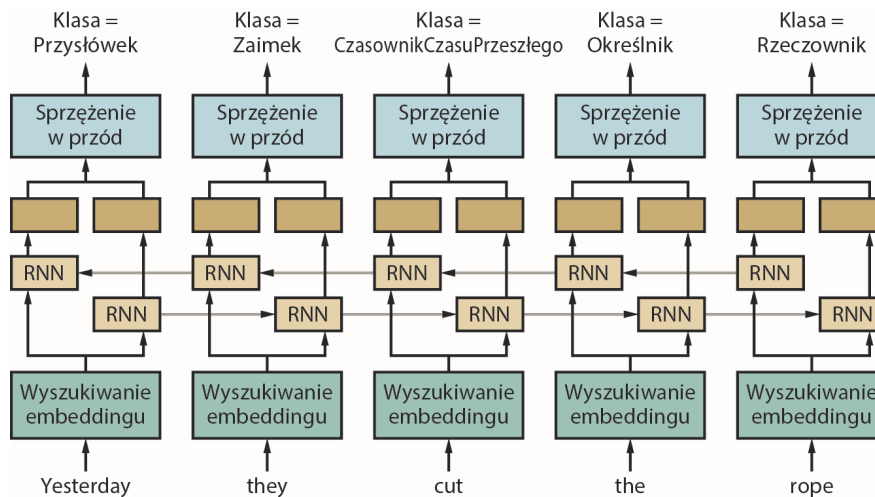
**RYСУNEK 24.1.** Embeddingi słów wyliczone przez algorytm GloVe wytrenowany na zbiorze liczącym 6 miliardów słów. Na rysunku widoczny jest efekt rzutowania 100-wymiarowych wektorów słów na dwuwymiarową płaszczyznę; wektory reprezentujące podobne słowa znajdują się blisko siebie



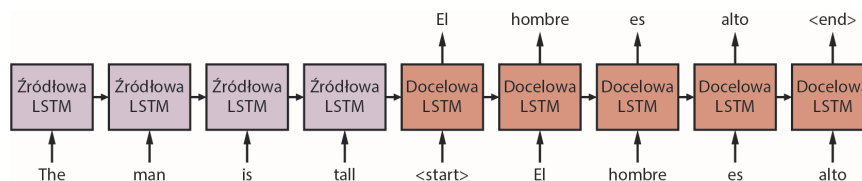
**RYСУNEK 24.2.** Model ze sprzężeniem w przód dla tagowania części mowy. Na podstawie składającego się z 5 słów okna jako wejścia model ten przewiduje kategorię gramatyczną (część mowy) dla środkowego słowa — w tym przypadku *cut*. Model zdolny jest do uwzględniania pozycji słowa w zdaniu, ponieważ każdy z 5 embeddingów mnożony jest przez różne części pierwszej ukrytej warstwy. Wartości parametrów dla embeddingów i dla trzech warstw są wynikiem zrównoleglonego treningu (*Yesterday they cut the rope* — „wczoraj [oni] przecięli linę”)



**RYСУNEK 24.3.** (a) Schematyczny diagram podstawowej sieci RNN, w której ukryta warstwa **z** zawiera rekurencyjne połączenia; symbol  $\Delta$  reprezentuje opóźnienie. Każdy wektor wejściowy **x** jest embeddingiem następnego słowa w zdaniu. (b) Ta sama sieć rozwinięta na trzy kroki czasowe, do postaci sieci ze sprzężeniem w przód. Zauważmy, że wagi współdzielone są przez wszystkie kroki czasowe

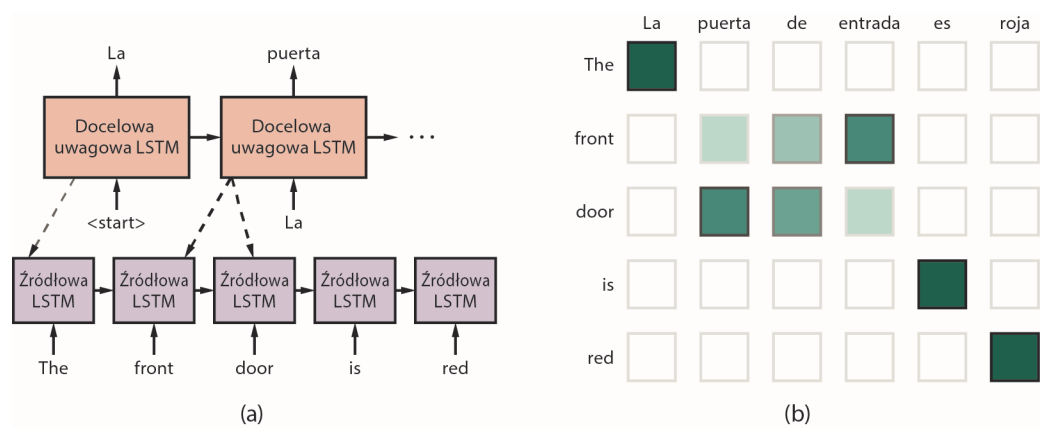


RYSUNEK 24.4. Dwukierunkowa sieć RNN dla tagowania części mowy

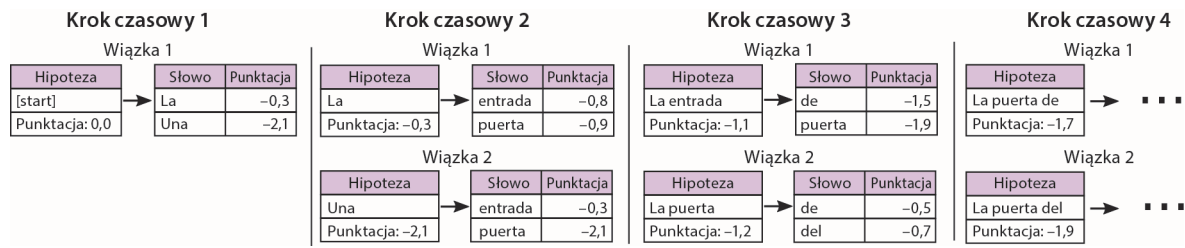


**RYSUNEK 24.5.** Podstawowy model „sekwencja na sekwencję”. Każdy blok reprezentuje jeden krok czasowy LSTM (dla prostoty pominęliśmy embeddingi i warstwy wyjściowe). W kolejnych krokach przekazujemy do sieci kolejne słowa zdania źródłowego *The man is tall* („Ten mężczyzna jest wysoki”), zakończonego ogranicznikiem <start>, oznaczającym, że sieć może rozpocząć generowanie zdania docelowego. Końcowy ukryty stan na końcu zdania źródłowego wykorzystywany jest jako ukryty stan na starcie zdania docelowego. Następnie każde słowo zdania docelowego w kroku  $t$  wykorzystywane jest jako wejście w kroku  $t + 1$  aż do momentu, gdy sieć wygeneruje znacznik <end> oznaczający, że generowanie zdania docelowego zakończyło się

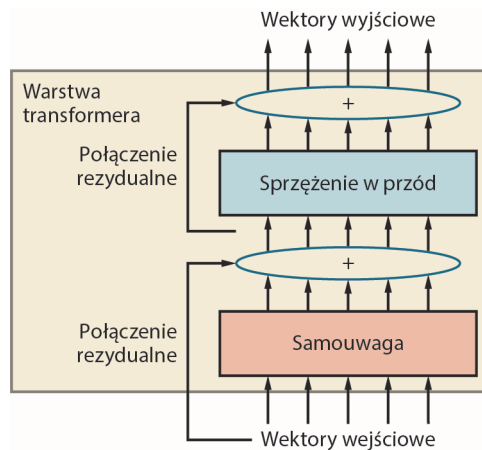




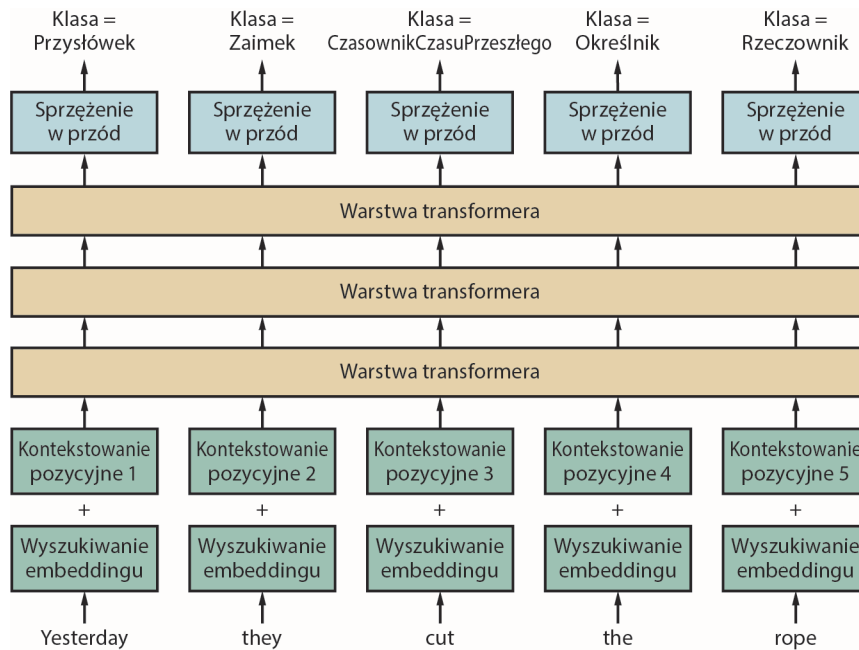
**RYSUNEK 24.6.** (a) Uwagowy model „sekwencja na sekwencję” dla tłumaczenia z języka angielskiego na hiszpański (*The front door is red* — „Drzwi frontowe są czerwone”). (b) Przykład uwagowej macierzy prawdopodobieństw dla dwujęzycznej pary zdań, ciemniejsze kwadraty reprezentują większe wartości  $a_{ij}$ . Wartości prawdopodobieństw uwagowych sumują się do 1 w każdej kolumnie



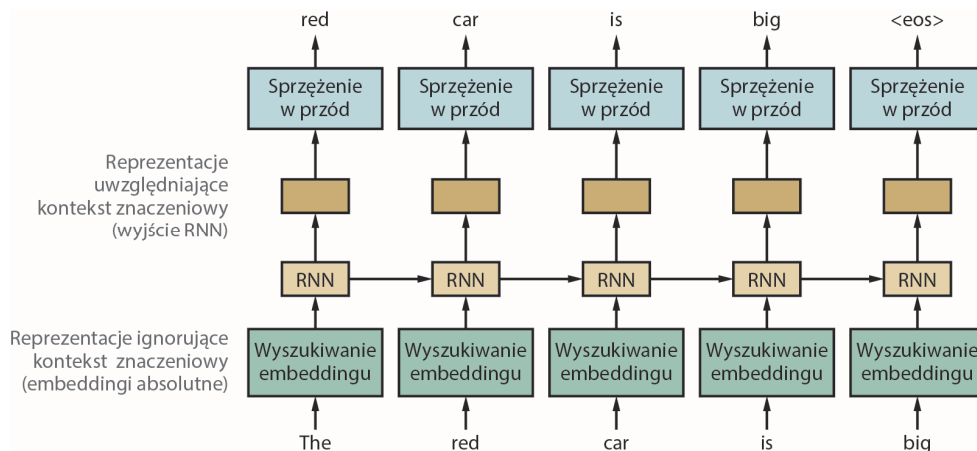
**RYSUNEK 24.7.** Wyszukiwanie skupione z rozmiarem wiązki  $b = 2$ . Punktacja każdego słowa równa jest logarytmicznemu prawdopodobieństwu wynikającemu z rozkładu softmax generowanego przez sieć RNN, a punktacja każdej hipotezy jest sumą punktacji poszczególnych słów. W kroku czasowym  $t = 3$  hipoteza o najwyższej punktacji *La entrada* może generować jedynie kontynuacje o niskim prawdopodobieństwie, więc plasuje się poza wiązką.



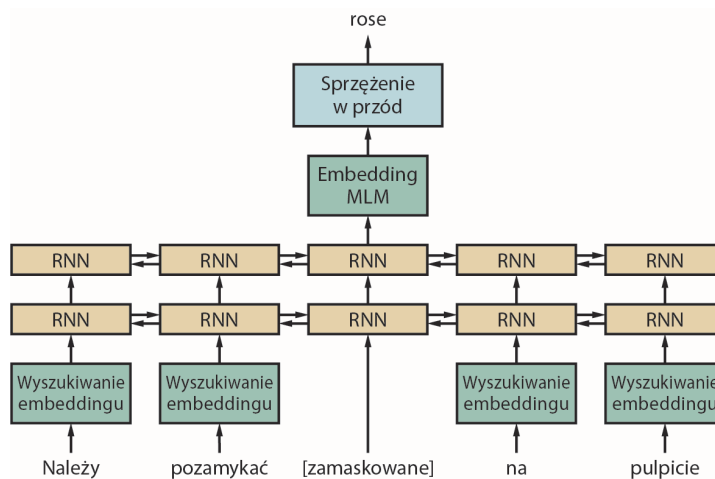
**RYSUNEK 24.8.** Jednowarstwowy transformer składający się z samouwagi, sieci ze sprzężeniem w przód i połączeń rezydualnych



**RYSUNEK 24.9.** Wykorzystywanie architektury transformera do tagowania części mowy.



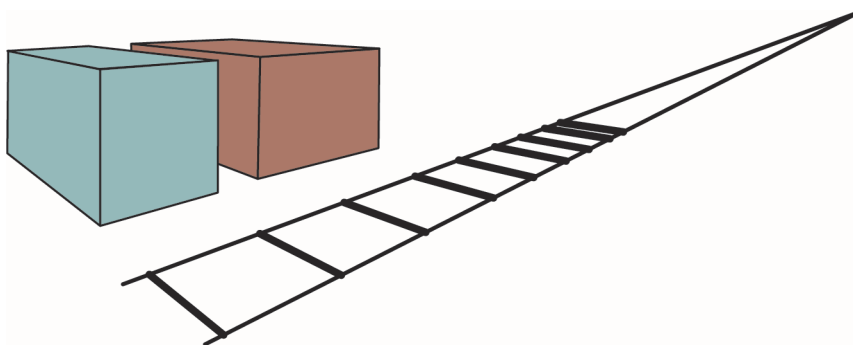
**RYSUNEK 24.10.** Trenowanie reprezentacji kontekstu zdaniowego w kierunku od lewej do prawej (*The red car is big* — „czerwony samochód jest duży”).



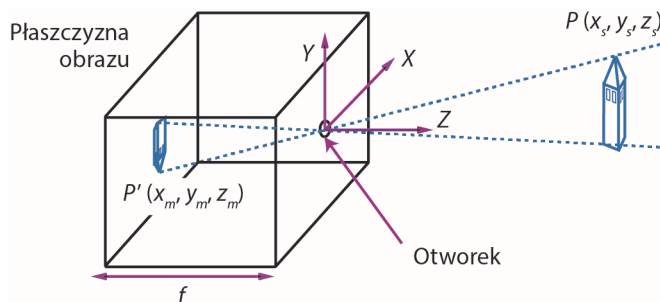
**RYSUNEK 24.11.** Modelowanie języka maskowanego: wstępny trening modelu dwukierunkowego — na przykład wielowarstwowej sieci RNN — przez maskowanie niektórych słów i następnie ich odgadywanie

## ROZDZIAŁ 25

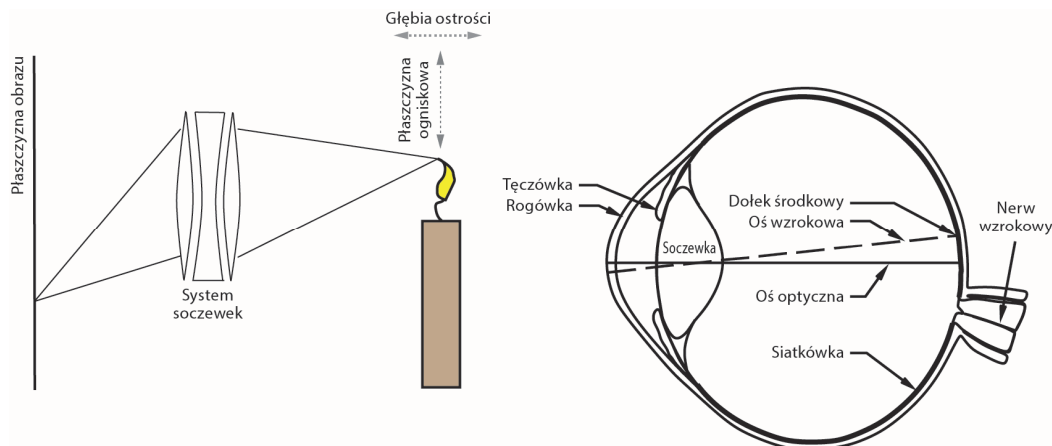
# WIDZENIE KOMPUTEROWE



**RYSUNEK 25.1.** Geometria sceny jest zniekształcona na jej obrazach. Linie w rzeczywistości równoległe wydają się zbiegać ze sobą, niczym na zdewastowanym torowisku. Budynki, które w rzeczywistości są prostopadłościanami, na obrazie mają kąty między ścianami różne od prostych



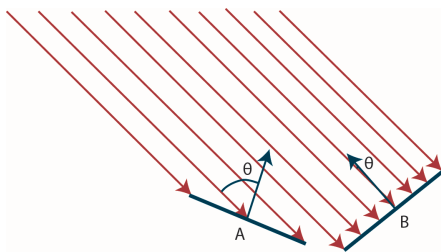
**RYSUNEK 25.2.** Światło, które przechodzi przez otworek z małego zakresu kierunków, trafia na elementy światłoczułe na tylnej ścianie kamery. Jeśli otworek jest wystarczająco mały, rezultatem jest skupiony obraz za otworkiem. Proces projekcji oznacza, że duże, odległe obiekty wyglądają jak podobne mniejsze pobliskie obiekty — punkt  $P'$  na płaszczyźnie obrazu mógł pochodzić z pobliskiej wieżyczki zabawkowej w punkcie  $P$  albo z odległej prawdziwej wieży w punkcie  $Q$



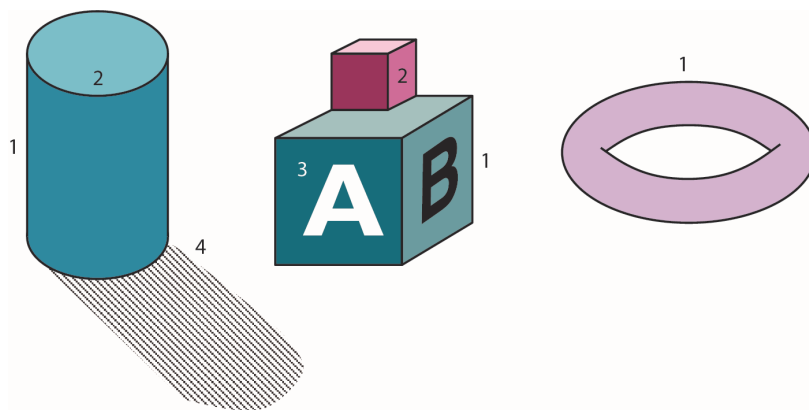
**RYSUNEK 25.3.** Soczewki zbierają światło opuszczające punkt na scenie (tu szczyt płomienia świecy) w różnych kierunkach i kierują całe światło tak, aby dotarło do pojedynczego punktu na płaszczyźnie obrazu. Promienie pochodzące z punktów sceny znajdujących się w pobliżu płaszczyzny ogniskowej — w zakresie głębi ostrości — będą prawidłowo skupiane. W kamerze elementy układu soczewek można przesuwac, w celu zmiany płaszczyzny ogniskowania, natomiast w oku efekt ten następuje w wyniku zmiany kształtu soczewki przez wyspecjalizowane mięśnie



**RYSUNEK 25.4.** Ilustracja różnych efektów oświetlenia. Na pojemniku ze stali nierdzewnej widoczne są odbłaski. Powierzchnie cebuli i marchewki są jasne i rozpraszające, ponieważ są skierowane w stronę światła. Cienie pojawiają się w tych punktach powierzchni, do których w ogóle nie dociera światło ze źródła. Wewnątrz czajnika znajdują się ciemne, rozpraszające powierzchnie, na które światło pada w kierunku stycznym (widoczne są też cienie). Zdjęcie: Ryman Cabannes/Image Professionals GmbH/Alamy Stock Photo



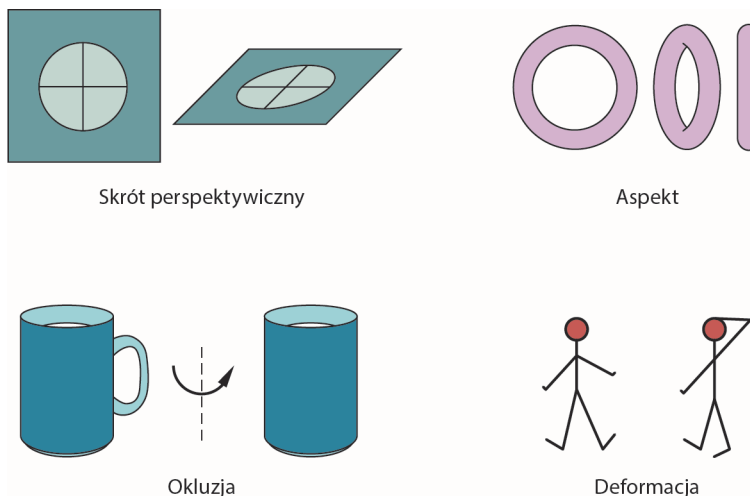
**RYSUNEK 25.5.** Dwa skrawki powierzchni oświetlane przez odległe źródło punktowe; strzałki reprezentują biegnące promienie świetlne. Skrawek A jest nachylony niemal stycznie do promieni ( $\theta$  jest bliski  $90^\circ$ ) i zbiera mało energii, ponieważ przecina mniej promieni świetlnych w przeliczeniu na jednostkę powierzchni. Skrawek B skierowany jest prostopadle do promieni ( $\theta$  jest bliski  $0^\circ$ ) i zbiera więcej energii



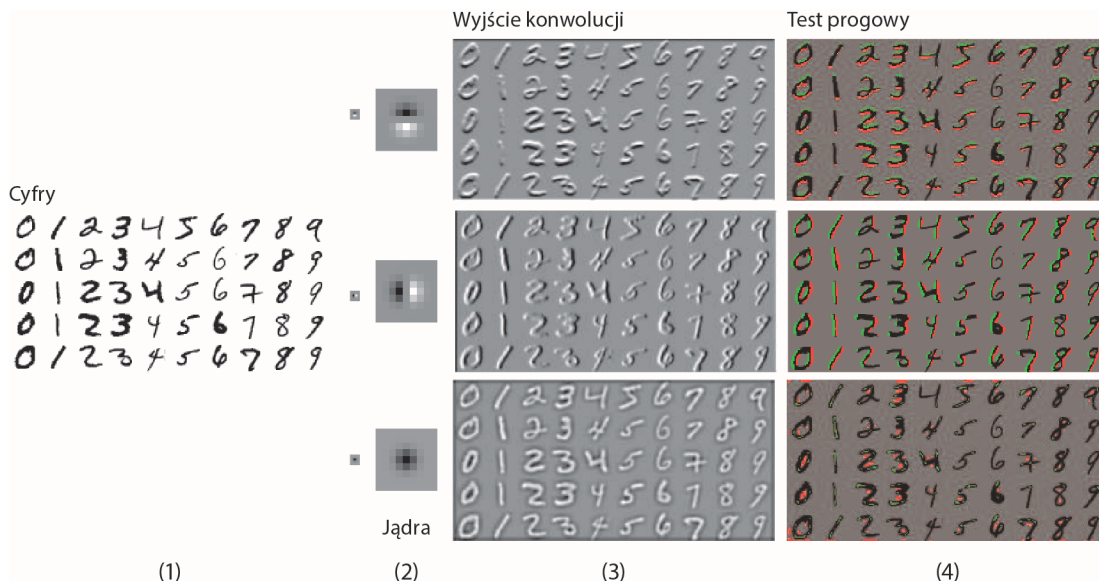
**RYSUNEK 25.6.** Różne rodzaje krawędzi: (1) nieciągłości głębokości; (2) nieciągłości orientacji powierzchni; (3) nieciągłości odbicia; (4) nieciągłości oświetlenia (cienie)



**RYSUNEK 25.10.** (a) Obraz oryginalny. (b) Kontury graniczne — im wyższa wartość  $P_b$ , tym ciemniejszy kontur. (c) Segmentacja odpowiadająca droboziarnistemu podziałowi obrazu na regiony. Regiony renderowane są w ich uśrednionych kolorach. (d) Segmentacja, odpowiadająca bardziej gruboziarnistemu podziałowi na regiony i w rezultacie z mniejszą liczbą regionów. (Zdjęcia dzięki uprzejmości Pablo Arbelaeza, Michaela Maire’a, Charlesa Fowlkesa i Jitendry Malika)

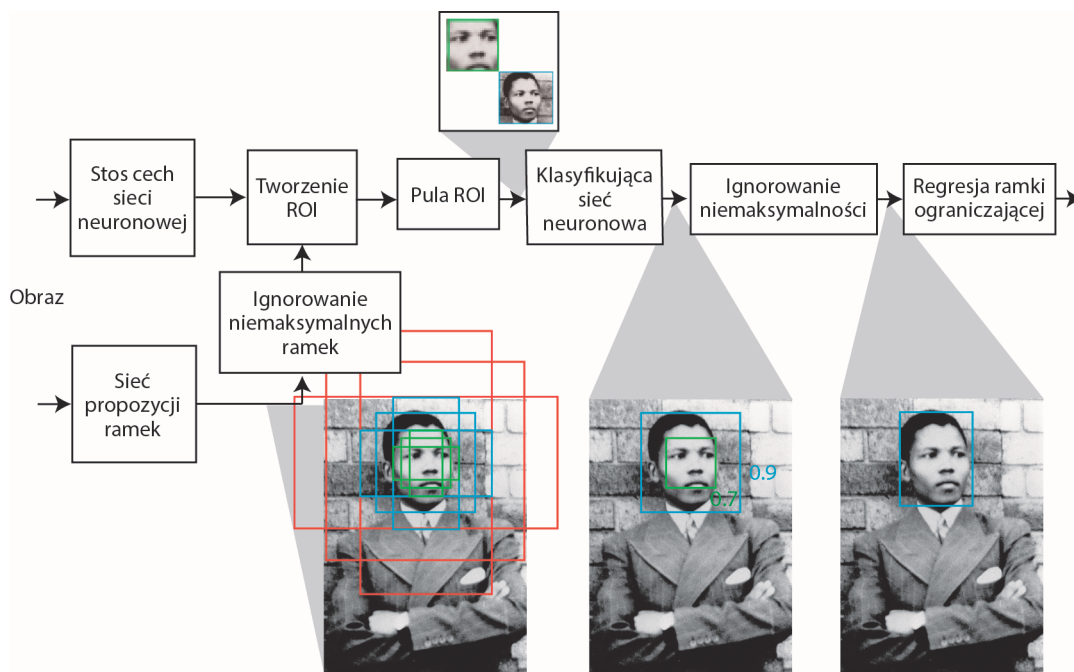


**RYSUNEK 25.11.** Różne istotne źródła zmienności wyglądu, sprawiającej, że różne obrazy tego samego obiektu wyglądają inaczej. Po pierwsze, elementy mogą być zniekształcone przez skrót perspektywiczny — jak kolistą powierzchnię oglądaną pod kątem, widoczna w lewej górnej części. Po drugie, obiekty oglądane z różnych kierunków mogą dość drastycznie zmieniać proporcje („aspekt”), jak trzy różne obrazy pączka w kształcie torusa w prawej górnej części. Po trzecie, okluzja może powodować niewidoczność pewnych obiektów lub ich części — z lewej strony u dołu obracany kubek zakrywa własne ucho (autookluzja). Po czwarte wreszcie, obiekty mogą drastycznie zmieniać swój kształt, jak sylwetka z prawej strony u dołu

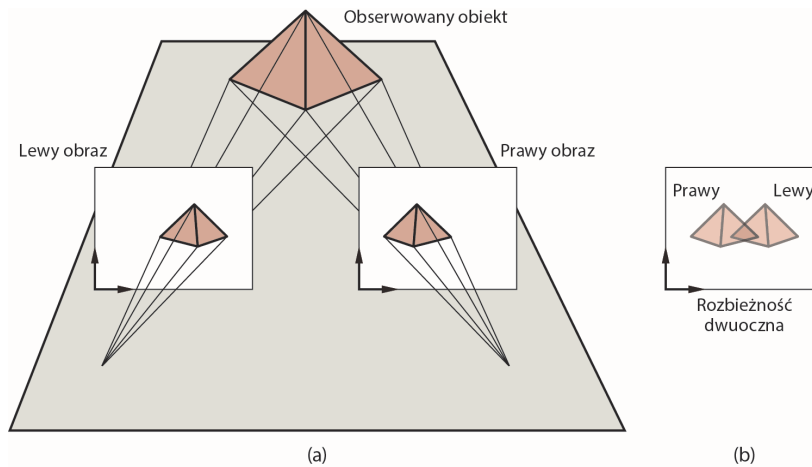


**RYSUNEK 25.12.** W kolejnych kolumnach: (1) Kilka obrazków ze zbioru MNIST. (2) Trzy jądra, pokazane w naturalnej wielkości i w powiększeniu, w celu pokazania zawartości: szarość oznacza zero, jasny odcień wartość dodatnią, ciemny odcień wartość ujemną. (3) Rezultaty zastosowania tych jąder do obrazów. (4) Piksele, dla których odpowiedź jest większa od ustalonego progu (kolor zielony) lub mniejsza od niego (kolor czerwony). Zauważmy, że daje to (kolejno od góry do dołu): detektor poziomego paska; detektor pionowego słupka; i (co trudniej spostrzec) detektor końca linii. Detektory te biorą pod uwagę kontrast paska, więc (na przykład) poziomy pasek, który jest jasny na górze i ciemny pod spodem, daje pozytywną odpowiedź (zielony kolor), a ten, który jest ciemny u góry i jasny pod spodem, daje odpowiedź ujemną (czerwony kolor). Nie jest to detekcja doskonała, ale jednak umiarkowanie efektywna

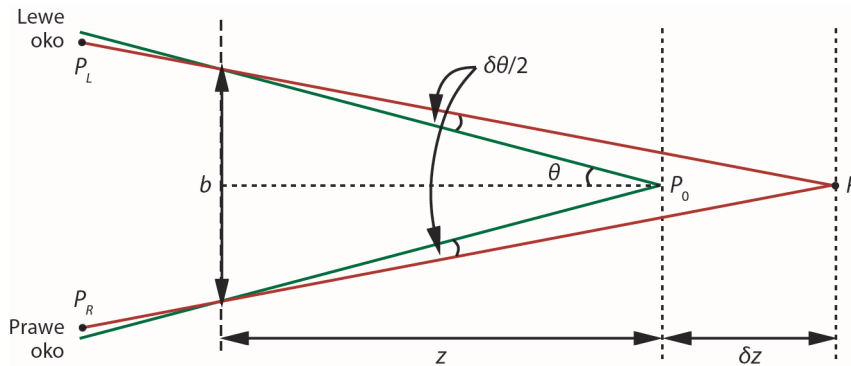




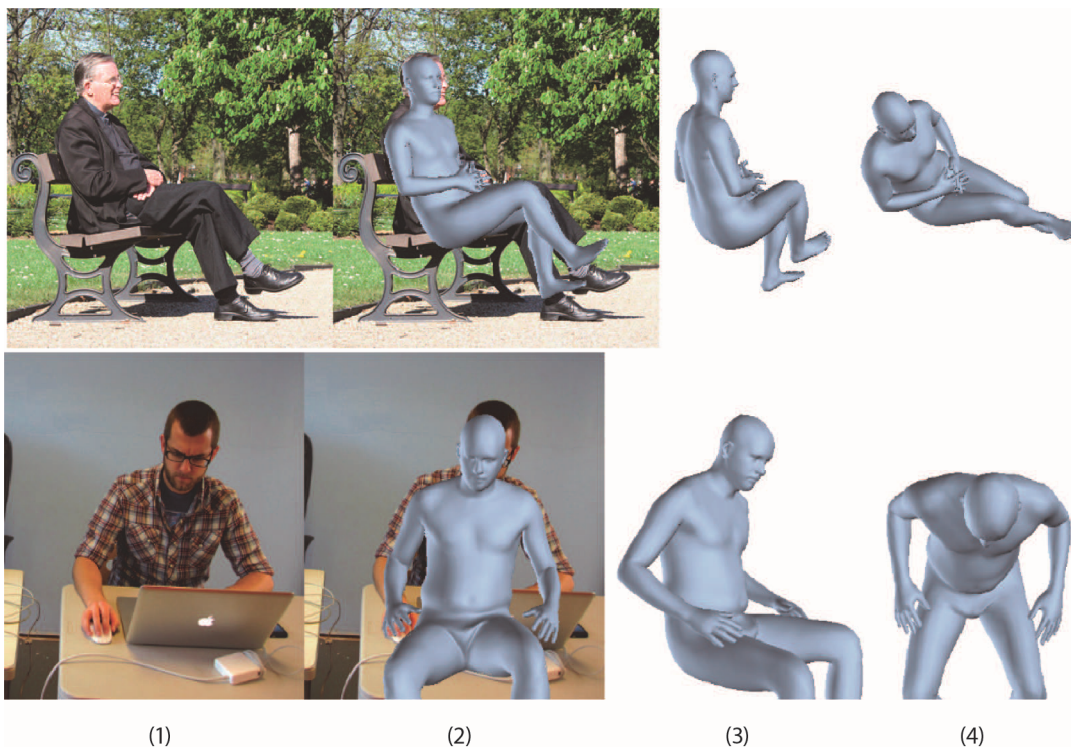
**RYSUNEK 25.13.** *Faster RCNN* wykorzystuje dwie sieci CNN. Do detektora obiektów trafia zdjęcie młodego Nelsona Mandeli. Jedna sieć ocenia „obiektość” kandydujących pól obrazu, zwanych „ramkami zakotwiczeń”, wyśrodkowanych w punktach gridu. Z każdym takim punktem związanych jest dziewięć ramek (trzy skale  $\times$  trzy proporcje). Dla danego obrazu wewnętrzna zielona ramka i zewnętrzna niebieska ramka to ramki, które zaliczyły test na obiektość. Druga ze wspomnianych sieci jest stosem cech i konstruuje reprezentację obrazu odpowiednią dla jego klasyfikacji. Ramki z najwyższym wskaźnikiem obiektości są wycinane z mapy cech, standaryzowane pod względem rozmiaru za pomocą poolingingu regionów zainteresowań (ROI) i przekazywane do klasyfikatora. Ramki niebieska i zielona zachodzą na siebie, więc w procesie ignorowania niemaksymalności zielona jest odrzucana, bo ma mniejszą punktację. Wreszcie, regresja ramki ograniczającej dopasowuje niebieską ramkę do twarzy. Oznacza to, że mimo stosunkowo zgrubnego charakteru próbkowania lokalizacji, skal i proporcji, klasyfikacja nie traci na dokładności. Zdjęcie: Sipa/Shutterstock



**RYSUNEK 25.14.** Przesunięcie kamery równoległe do płaszczyzny obrazu powoduje, że cechy obrazu poruszają się w płaszczyźnie kamery. Wynikająca z tego różnica pozycji jest wskazówką odnośnie głębi. Jeśli nałożymy lewy i prawy obraz na siebie, jak w (b), zobaczymy rozbieżność



**RYSUNEK 25.15.** Zależność między rozbieżnością a głębią w stereopsji. Środki rzutowania dwojga oczu oddległe są o  $b$ , a osie optyczne przecinają się w punkcie zafiksowania  $P_0$ . Punkt  $P$  sceny rzutowany jest na punkty  $P_L$  i  $P_R$  dla lewego i prawego oka. Rozbieżność kątowa między nimi wynosi  $\delta\theta$  (na diagramie pokazana jako suma dwóch kątów  $\delta\theta/2$ )



**RYСУNEK 25.16.** Rekonstrukcja postaci ludzkiej 3D na podstawie jednego obrazu jest teraz wykonalna w praktyce — każdy z dwóch rzędów przedstawia wynik takiej rekonstrukcji. Wykonalność takich rekonstrukcji wynika z możliwości estymacji położenia stawów, rzeczywistych kątów zgięcia w stawach, kształtu i pozycji ciała względem płaszczyzny obrazu. W kolejnych kolumnach obrazu: (1) przedmiotowe zdjęcie (2) wynik rekonstrukcji postaci nałożony na oryginalny obraz (3) i (4) inne widoki zrekonstruowanej postaci. Wiele różnych widoków ciała znacznie utrudnia ukrycie błędów rekonstrukcji. Rysunek dzięki uprzejmości Angjoo Kanazawy, wyprodukowany przez system opisany w (Kanazawa i in, 2018a)



**RYSUNEK 25.17.** To samo działanie może wyglądać zupełnie inaczej, a różne działania mogą wyglądać podobnie. Przedstawione przykłady zaczerpnięte zostały ze zbioru danych kolekcjonujących obrazy dotyczące naturalnych zachowań, etykiety tych obrazów zostały skonstruowane przez ludzi, nie są wynikiem przewidywań algorytmicznych. **Część górna:** przykłady obrazów etykietowanych jako „otwieranie lodówki” — niektóre pokazane w zbliżeniu, inne z pewnego dystansu. **Część dolna:** obrazy etykietowane jako „wyjmij coś z lodówki”. Zwróćmy uwagę, że w obu przypadkach dłoń osoby znajduje się w pobliżu drzwi, lecz w drugim przypadku ich nie dotyka. Wykrycie tej różnicy wymaga dość subtelного osądu. Rysunek dzięki uprzejmości Davida Fouhey’a, zaczerpnięty ze zbioru danych opisanego w (Fouhey i in., 2018)



**RYSUNEK 25.18.** To, co uważamy za akcję, zależy jest od skali czasu, w której akcję tę rozpatrujemy. Czynność, którą przedstawia pojedyncze zdjęcie **w górnym rzędzie**, można określić jako (po prostu) otwieranie lodówki — nie sposób wywnioskować ze zdjęcia, by osoba otwierająca drzwi była zainteresowana jakimś konkretnym produktem. Gdybyśmy jednak sfilmowali tę scenę i przedstawili kilka klatek z tego krótkiego klipu (jak **w środkowym rzędzie**), zobaczylibyśmy, że wspomniana czynność to wyjmowanie butelki z mlekiem. A gdy klip będzie dłuższy (jak **w rzędzie dolnym**), stanie się jasne, że osoba przyrządza sobie przekąskę. Zauważ, że rysunek ilustruje pewien aspekt kompozycyjności zachowań: wyjęcie mleka z lodówki może być częścią przygotowywania przekąski, a otwarcie lodówki jest konieczne do wyjęcia z niej butelki z mlekiem. Rysunek dzięki uprzejmości Davida Fouhey’a, zaczerpnięty ze zbioru danych opisanego w (Fouhey i in., 2018)



*A baby eating a piece  
of food in his mouth*  
(Niemowlę jedzące  
ustami porcję pożywienia)



*A young boy eating  
a piece of cake*  
(Mały chłopczyk spożywający  
kawałek ciastka)



*A small bird is perched  
on a branch*  
(Mały ptaszek  
siedzi na gałęzi)



*A small brown bear  
is sitting in the grass*  
(Mały niedźwiadek  
brunatny siedzi na trawie)

**RYSUNEK 25.19.** Zautomatyzowane systemy podpisywania obrazów mogą dawać podpisy zarówno trafne, jak i bezsensowne. Dwa podpisy z lewej strony zdecydowanie dobrze opisują odpowiednie obrazy, chociaż sformułowanie „jedzenie ... w ustach” jest raczej karykaturalne (notabene tego rodzaju niezgrabności były typowe dla wczesnych systemów podpisywania, opartych na modelach językowych budowanych na bazie sieci RNN). Z dwóch opisów z prawej strony można wywnioskować, że system podpisywania nie posiada raczej żadnej wiedzy na temat wiewiórek i próbuje zgadywać na podstawie kontekstu; można również przypuszczać, że system nie zauważył, iż obie wiewiórki oddają się konsumpcji przysmaków. Zdjęcia udostępnione przez: geraine/Shutterstock; ESB Professional/Shutterstock; BushAlex/Shutterstock; Maria.Tem/Shutterstock. Pokazane zdjęcia są podobne, ale nie identyczne z oryginalnymi, dla których wygenerowano podpisy; oryginalne zdjęcia można znaleźć w (Aneja i in., 2018)



Pyt.: *What is the cat wearing?*  
(Co ma na sobie kot?)

Odp.: *Hat* (Kapelusz)



Pyt.: *What is the weather like?*  
(Jaka jest pogoda?)

Odp.: *Rainy* (Deszczowa)



Pyt.: *What surface is this?*  
(Jaka to nawierzchnia?)

Odp.: *Clay* (Gliniana)



Pyt.: *What toppings are on the pizza?*  
(Jakie są dodatki do tej pizzy?)

Odp.: *Mushrooms* (Grzyby)



Pyt.: *How many holes are in the pizza?*  
(Ile dziur jest w tej pizzy?)

Odp.: 8



Pyt.: *What letter is on the racket?*  
(Jaka litera widnieje na rakiecie?)

Odp.: w



Pyt.: *What color is the right front leg?*  
(Jakiego koloru jest prawa przednia noga?)

Odp.: *Brown* (Brązowego)



Pyt.: *Why is the sign bent?*  
(Dlaczego ten znak jest wygięty?)

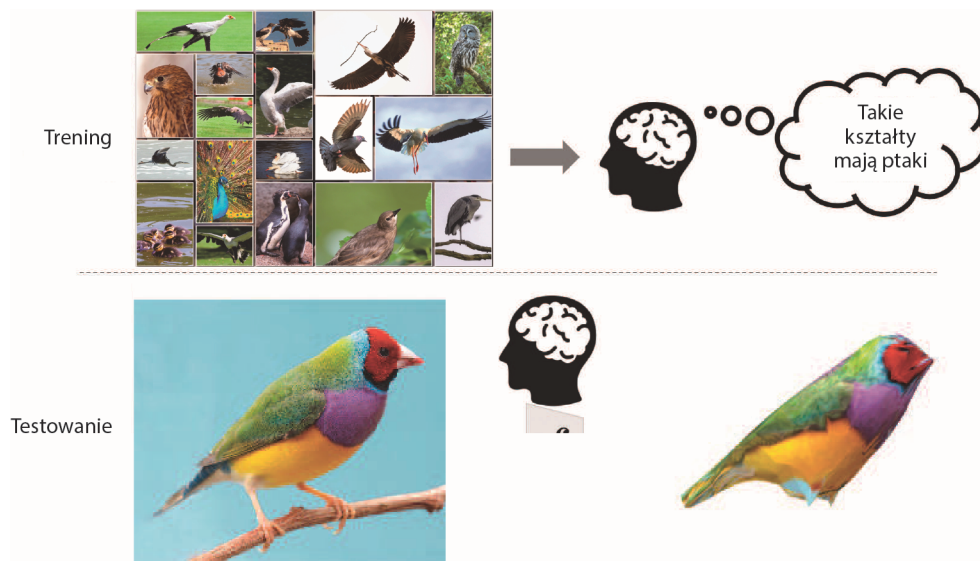
Odp.: *It's not* (Nie jest)

**RYSUNEK 25.20.** Wizualne systemy odpowiadania na pytania generują odpowiedzi (zwykle wybierane z gotowych zestawów) na pytania dotyczące obrazów, formułowane w języku naturalnym. **Część górna:** system daje całkiem sensowne odpowiedzi na dość trudne pytania dotyczące obrazu. **Część dolna:** odpowiedzi są mniej satysfakcjonujące, ale wciąż poprawne. Jeśli na przykład system ma określić liczbę dziur w pizzy, to skazany jest wyłącznie na zgadywanie, ponieważ nie rozumie, co można uznać za dziurę i generalnie ma duże trudności z liczeniem. Podobnie system uważa, że kocia noga jest koloru brązowego, ponieważ nie jest w stanie prawidłowo jej zlokalizować; arbitralnie więc domniemywa jej położenie, trafiając niestety na brązowe tło. Zdjęcia udostępnione przez: (część górna) Tobyanna/Shutterstock; 679411/Shutterstock; ESB Professional/Shutterstock; Afryka Studio/Shutterstock; (część dolna) Stuart Russell; Maxisport/Shutterstock; Chendongshan/Shutterstock; Scott Biales DitchTheMap/Shutterstock. Przedstawione zdjęcia są podobne do oryginalnych, do których zastosowano system odpowiadania, ale nie identyczne z nimi. Oryginalne zdjęcia znaleźć można w Goyal i in. (2017)





**RYSUNEK 25.21.** Modele 3D placów budowy tworzone są przy użyciu zarówno algorytmów „struktury z ruchu” (ang. SfM — *structure-from-motion*), jak i algorytmów analizy wielowidokowej. Pomagają firmom budowlanym koordynować prace nad dużymi budynkami, przez porównywanie modelu 3D aktualnej konstrukcji z planami budowlanymi. **Po lewej:** wizualizacja modelu geometrycznego uchwyconego przez drony. Zrekonstruowane punkty 3D są renderowane w kolorze, więc z wizualizacji można odczytać dotychczasowe postępy prac (zauważmy częściowo ukończony budynek z dźwigiem). Małe piramidki pokazują pozycje drona podczas robienia zdjęć i składają się na wizualizację trajektorii jego lotu. **Po prawej:** takie systemy są faktycznie wykorzystywane przez ekipy budowlane; zespół na spotkaniu koordynacyjnym konfrontuje aktualny stan robót z planami na podstawie wizualizacji modelu. Rysunek dzięki uprzejmości Dereka Hoiema, Mani Golparvar-Fard i Reconstruct. Wyprodukowany przez komercyjny system opisywany na blogu [medium.com/reconstruct-inc](https://medium.com/reconstruct-inc)

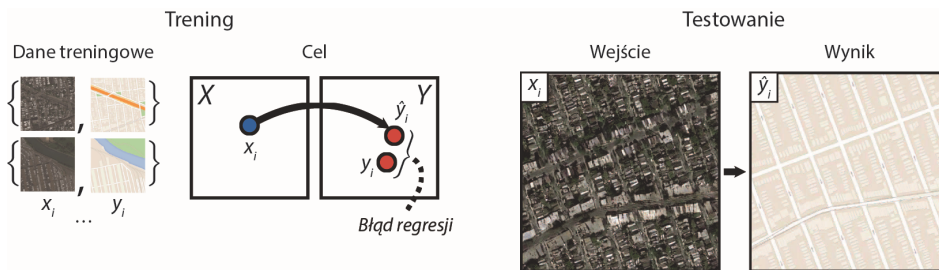


**RYSUNEK 25.22.** Jeśli widziałeś wiele zdjęć z jakiejś kategorii — na przykład ptaków (**część górna**) — możesz użyć swojej wiedzy do odtworzenia wyglądu 3D rzeczywistego obiektu (ptaka) na podstawie tylko jednego, nieznanego Ci dotychczas widoku (**część dolna**). Musisz jednak mieć pewność, że wszystkie obiekty przedstawione na wspomnianych zdjęciach mają zbliżoną geometrię — zdjęcie strusia nie pomoże Ci więc w rekonstrukcji wyglądu wróbla — choć metody klasyfikacji potrafią właściwie traktować takie różnice. Na podstawie wielu widoków możesz oszacować rozkład tekstury na rzeczywistym obiekcie i tym samym uzupełnić ewentualne jej braki (część dolna). Rysunek dzięki uprzejmości Angjoo Kanazawy. Wyprodukowany przez system opisany w Kanazawa i in. (2018b). Górne zdjęcie: Satori/123RF; dolne zdjęcie po lewej: Four Oaks/Shutterstock

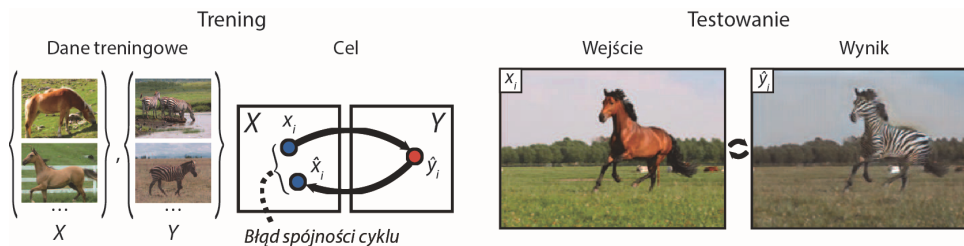




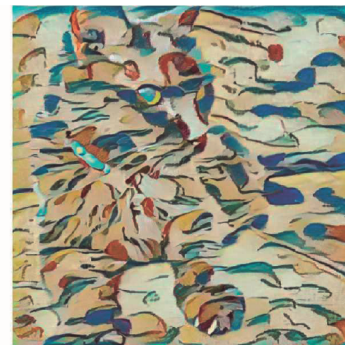
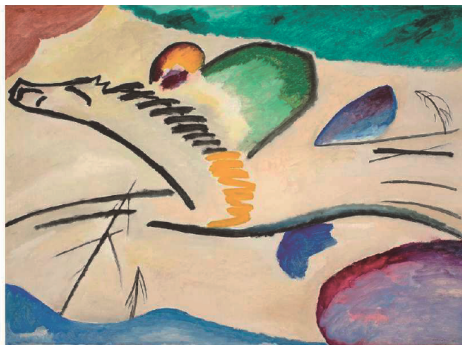
**RYSUNEK 25.23. Lewa strona:** oryginalny obraz sceny. **Prawa strona:** w obraz sceny został wkomponowany obiekt grafiki komputerowej; jego oświetlenie oraz rzucane przez niego cienie wyglądają tak, jakby był prawdziwy. Spreparowany obraz wygląda przekonująco, w spreparowanym oświetleniu lub spreparowanych cieniach pojawiają się drobne błędy — przeciętny widz niebędący ekspertem raczej ich nie zauważy. Rysunek dzięki uprzejmości Kevina Karscha. Stworzony przez system opisany w Karsch i in. (2011)



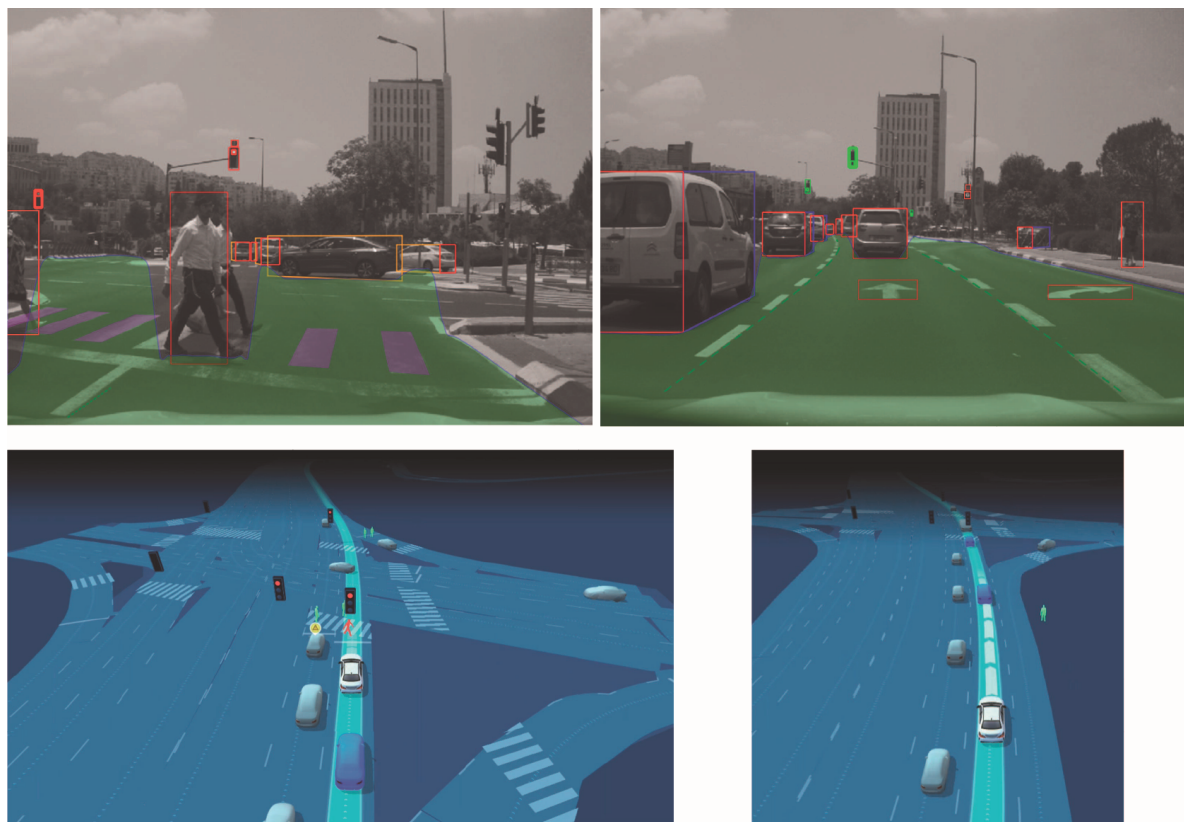
**RYSUNEK 25.24. Tłumaczenie obrazów:** wejście stanowią zdjęcia lotnicze i odpowiadające im kwadraty mapy, celem treningu jest nauczenie systemu sporządzania kwadratów mapy dla nowych zdjęć lotniczych (możliwe jest także trenowanie systemu w przeciwnym kierunku — odtwarzania widoku na podstawie odpowiadającego mu kwadratu mapy). Trenowanie sieci odbywa się przez porównywanie  $\hat{y}_i$  (wyjścia dla przykładu  $x_i$  typu  $X$ ) z prawidłowym wyjściem  $y_i$  typu  $Y$ . Na etapie testowania sieć musi produkować nowe obrazy typu  $Y$  odpowiadające otrzymywanym obrazom typu  $X$ . Rysunek dzięki uprzejmości Phillipa Isoli, Jun-Yan Zhu i Alexeia A. Efrosa. Stworzony przez system opisany w Isola i in. (2017). Dane mapy © 2019 Google



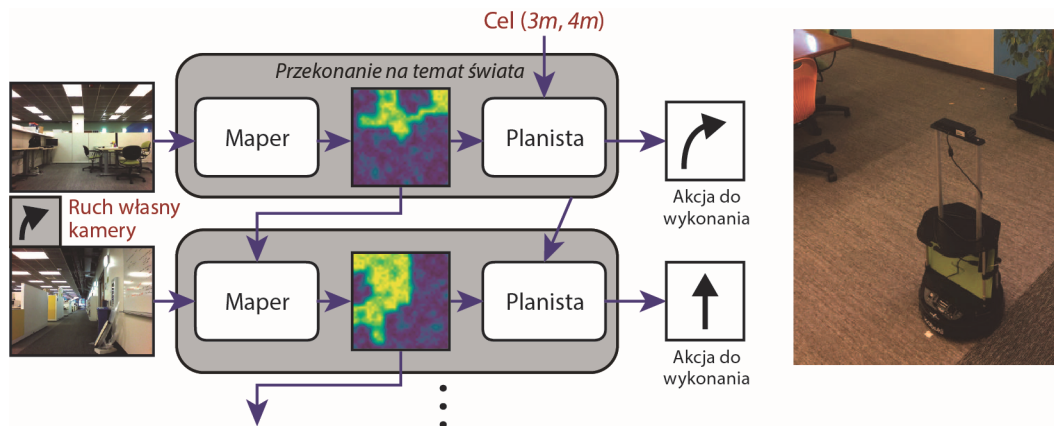
**RYСУNEK 25.25.** Tłumaczenie obrazów na podstawie dwóch oddzielnych kolekcji — typu  $X$  i typu  $Y$  — bez łączenia obrazów w pary; na rysunku typ  $X$  to konie, a typ  $Y$  to zebry. Ponieważ nie jest możliwe trenowanie systemu za pomocą par odpowiedników „koń-zebra”, trening przeprowadza się za pomocą dwóch przeciwnastawnych mapowań, najpierw z typu  $X$  na typ  $Y$ , potem z typu  $Y$  na typ  $X$ : dla danego konia  $x_i$  pierwsza sieć typuje odpowiadającą mu zebrową  $\hat{y}_i$ , następnie druga sieć typuje konia  $\hat{x}_i$  odpowiadającego zebrowi  $\hat{y}_i$ . W bezbłędnie typującym systemie  $\hat{x}_i$  byłoby identyczne z  $x_i$ , więc różnica między  $\hat{x}_i$  a  $x_i$  powinna stanowić kryterium trenowania obu wspomnianych sieci — trening ten powinien doprowadzić do zniwelowania tej różnicy, czyli *zamknięcia cyklu*  $X \rightarrow Y \rightarrow X$ . Tak wytrenowane sieci zdolne są radzić sobie nawet z bardzo zaawansowanymi transformacjami obrazów. Rysunek dzięki uprzejmości Aleksieja A. Efrosa; patrz Zhu i in. (2017). Zdjęcie biegnącego konia: Justyna Furmanczyk Gibaszek/Shutterstock



**RYСУNEK 25.26.** Przeniesienie stylu — *treść* zdjęcia (kot) zostaje połączona ze *stylem* abstrakcyjnego malowidła, co w wyniku daje nowy obraz kota, renderowany w stylu tegoż malowidła. Malowidło to *Lyrisches* Wassilija Kandinsky'ego (domena publiczna); kot ma na imię Cosmo



**RYSUNEK 25.28.** Funkcjonowanie kamery Mobileye autonomicznego samochodu. **Górny rząd:** dwa zdjęcia z przedniej kamery wykonane w odstępie kilku sekund; zielony obszar to wolna przestrzeń, do której pojazd mógłby fizycznie przemieścić się w najbliższej przyszłości. Wokół rozpoznanych obiektów 3D wyświetlane są obwiednie, w kolorach wskazujących strony obiektów (czerwony dla tyłu, niebieski dla prawej strony, żółty dla lewej strony i zielony dla przodu). Do wspomnianych obiektów zaliczają się pojazdy, piesi, wewnętrzne krawędzie granic pasa ruchu (niezbędne dla kontroli bocznej), inne elementy oznakowania poziomego, przejścia dla pieszych, znaki drogowe i sygnalizatory świetlne. Nie są wyróżnione zwierzęta, słupy, pachołki, chodniki, balustrady i inne przedmioty, które ewentualnie mogłyby się znaleźć w pobliżu (np. kanapa, która wypadła z tyłu ciężarówki). Każdy obiekt jest następnie oznaczany pozycją 3D i prędkością. **Dolny rząd:** pełny model fizyczny środowiska, wyrenderowany na podstawie wykrytych obiektów. (Pokazano jedynie wizualne wyniki systemu Mobileye, bez objaśnień tekstowych, m.in. pozycji i prędkości obiektów). Zdjęcia dzięki uprzejmości Mobileye



**RYСУNEK 25.29. Po lewej:** problem nawigacji rozwiązuje się poprzez dekompozycję na dwa podproblemy — mapowanie i planowanie. Z każdym kolejnym krokiem czasowym informacje z czujników wykorzystywane są do przyrostowego budowania niepewnego modelu świata. Model ten, wraz ze specyfikacją celu, jest przekazywany do planisty wyznaczającego kolejną akcję, jaką powinien wykonać robot w dążeniu do osiągnięcia celu. Modele świata mogą być czysto geometryczne (jak w klasycznym systemie SLAM), semantyczne (tworzone przez uczenie), a nawet topologiczne (oparte na punktach orientacyjnych). **Po prawej:** nawigujący robot. Rysunek i zdjęcie dzięki uprzejmości Saurabha Gupty

## ROZDZIAŁ 26

# ROBOTYKA

---



(a)



(b)

**RYSUNEK 26.1.** (a) Ramię robota przemysłowego z niestandardowym efekтором końcowym. Obraz udostępniony przez Macor/123RF. (b) Ramię robota Kinova® JACO® Assistive Robot zamontowane na wózku inwalidzkim. Kinova i JACO są znakami towarowymi firmy Kinova, Inc.

---



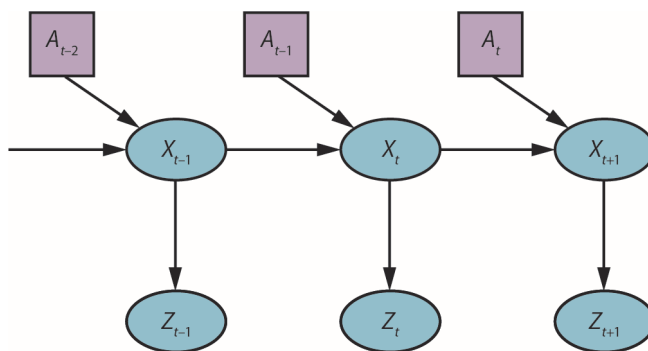


(a)

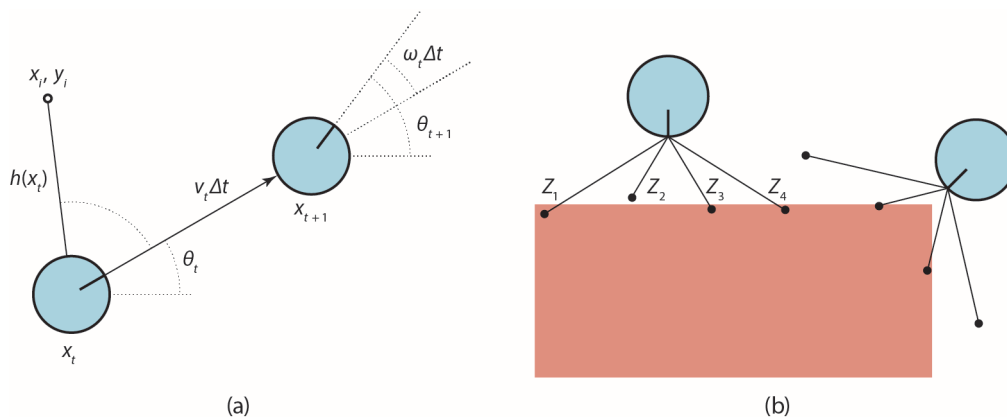


(b)

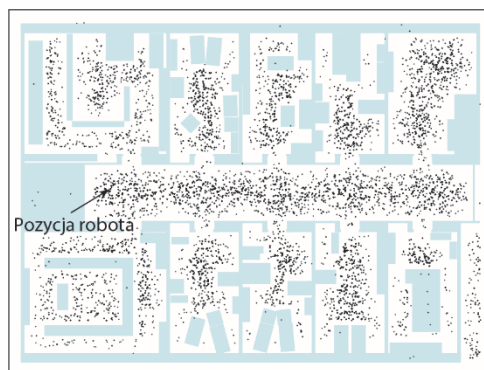
**RYСУNEK 26.2.** (a) Łazik Curiosity NASA wykonuje selfie na Marsie. Zdjęcie dzięki uprzejmości NASA. (b) Dron Skydio towarzyszący rodzinie podczas przejażdżki rowerowej. Zdjęcie dzięki uprzejmości Skydio



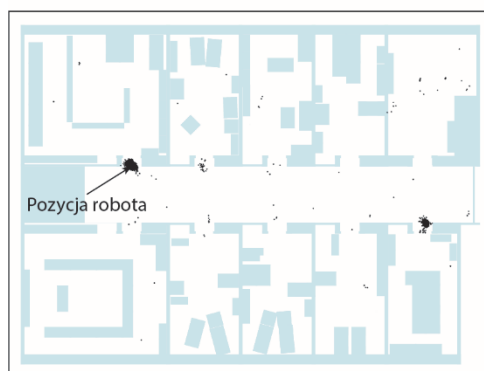
**RYСУNEK 26.4.** Percepcję robota można postrzegać jako wnioskowanie temporalne na podstawie sekwencji akcji i pomiarów, co ilustruje ta dynamiczna sieć decyzyjna



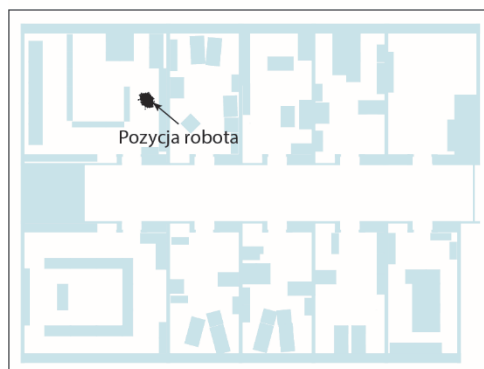
**RYСУNEK 26.5.** (a) Uproszczony model kinematyczny robota mobilnego. Robot jest przedstawiony jako okrąg z wewnętrzną linią promienia, oznaczającą kierunek do przodu. Na stan  $\mathbf{x}_t$  składa się para współrzędnych  $(x_t, y_t)$  (nie pokazano ich na rysunku) i orientacji  $\theta_t$ . Nowy stan  $x_{t+1}$  uzyskuje się przez aktualizację w pozycji  $v_t \Delta t$  i w orientacji  $\omega_t \Delta t$ . Pokazano również punkt orientacyjny w  $(x_i, y_i)$  obserwowany w czasie  $t$ . (b) Model czujnika ze skanowaniem odległości. Widoczne są dwie możliwe pozycje robota dla danego zakresu skanowania ( $z_1, z_2, z_3, z_4$ ). Dla pozycji po lewej wygenerowanie skanu jest o wiele bardziej prawdopodobne niż dla pozycji po prawej



(a)



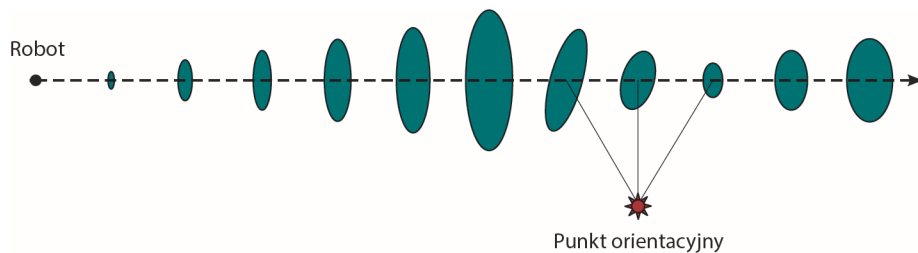
(b)



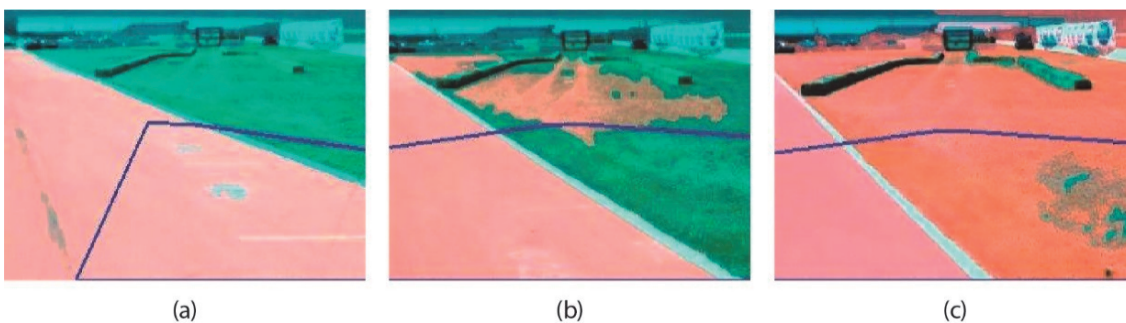
(c)

**RYSUNEK 26.6.** Lokalizacja Monte Carlo, algorytm filtrowania cząstek dla autolokalizacji robotów mobilnych. (a) Początkowa, globalna niepewność. (b) Aproksymowana bimodalna niepewność po nawigowaniu w (symetrycznym) korytarzu. (c) Niepewność unimodalna po wejściu do pokoju i stwierdzeniu, że jest on oddzielony od innych pomieszczeń

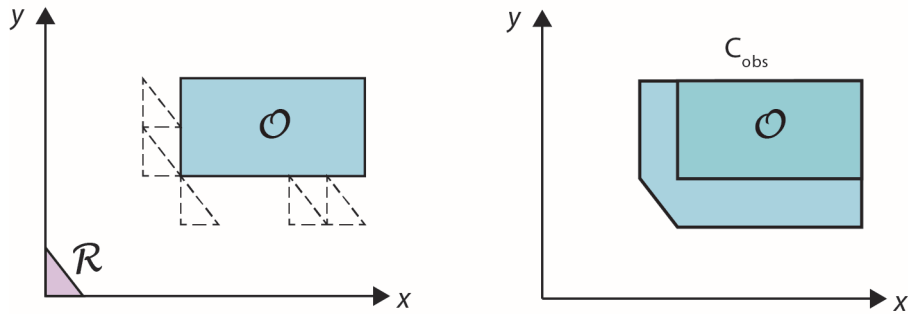




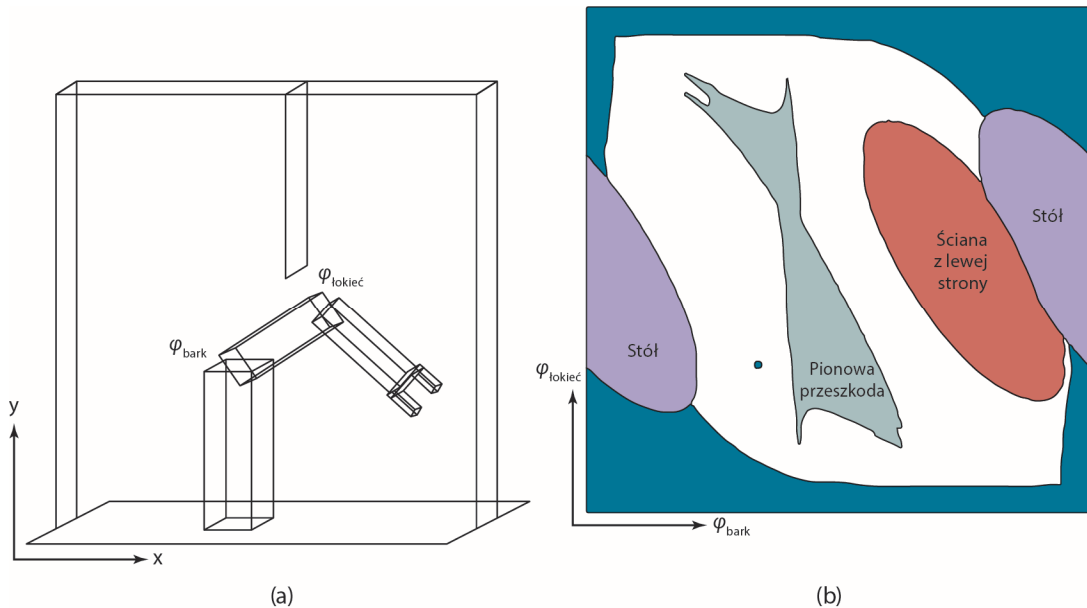
**RYSUNEK 26.8.** Lokalizacja przy użyciu rozszerzonego filtru Kalmana. Robot porusza się po linii prostej. W miarę postępu robota zwiększa się niepewność estymacji lokalizacji, co ilustrują elipsy błędów. Kiedy robot obserwuje punkt orientacyjny o znanej pozycji, niepewność się zmniejsza



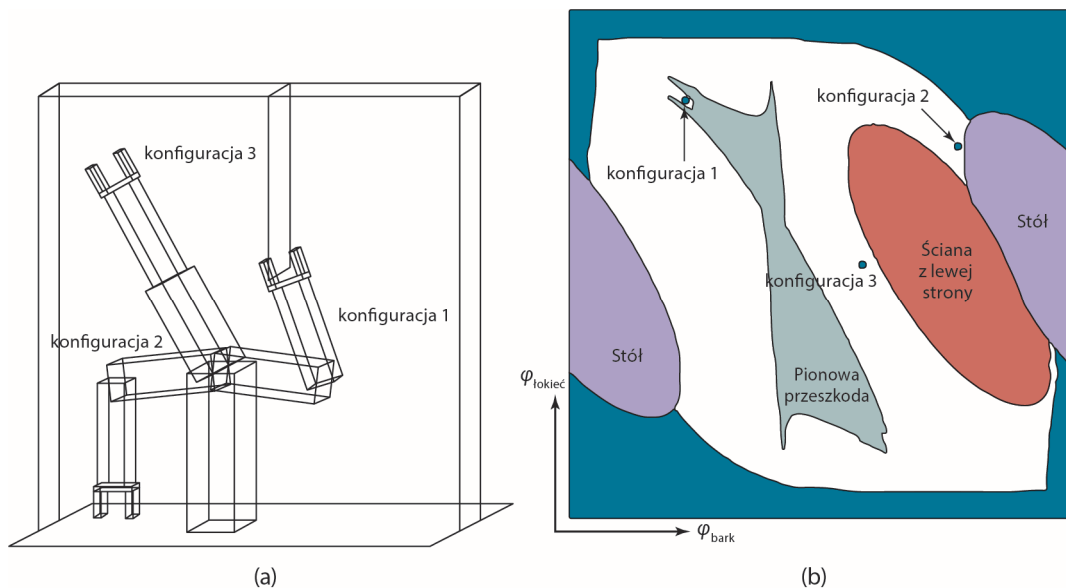
**RYSUNEK 26.9.** Sekwencja wyników klasyfikacji „powierzchnia zdalna do jazdy” z wykorzystaniem widzenia adaptacyjnego. (a) Tylko droga jest klasyfikowana jako zdalna do jazdy (obszar różowy). Niebieska linia w kształcie litery V pokazuje, dokąd zmierza pojazd. (b) Pojazd otrzymuje polecenie zjechania z drogi, a klasyfikator zaczyna klasyfikować część trawnika jako zdalną do jazdy. (c) Pojazd zaktualizował swój model „powierzchni zdalnych do jazdy” tak, aby uwzględnił i trawę, i drogę. Rysunek dzięki uprzejmości Sebastiana Thruna



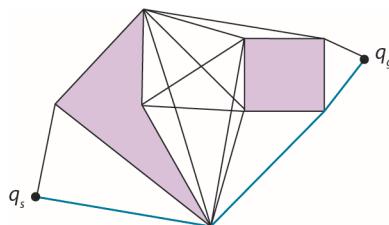
**RYСУNEK 26.10.** Prosty trójkątny robot, który potrafi się przesuwać i musi omijać prostokątną przeszkodę. Po lewej stronie widoczna jest jego przestrzeń robocza, po prawej przestrzeń konfiguracyjna



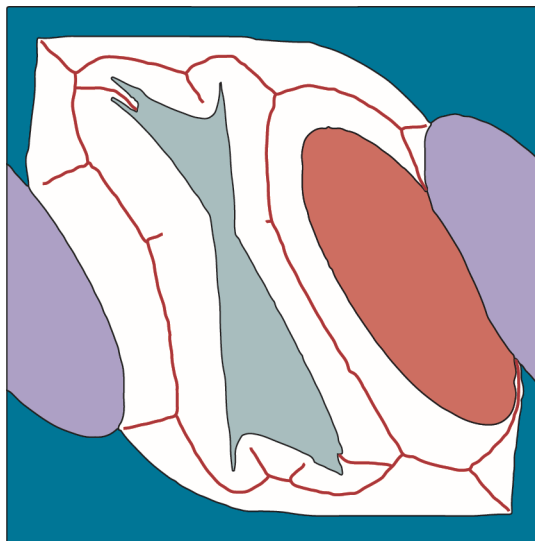
**RYСУNEK 26.11.** (a) Reprezentacja przestrzeni roboczej ramienia robota z dwoma stopniami swobody. Przestrzeń ta to skrzynka z płaską przeszkodą zwisającą z sufitu. (b) Przestrzeń konfiguracyjna tego samego robota — tylko białe obszary zawierają konfiguracje wolne od kolizji. Kropka na tym schemacie odpowiada konfiguracji robota pokazanej w lewej części



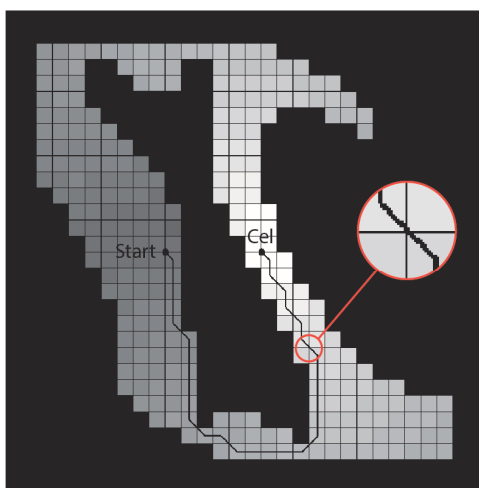
RYSUNEK 26.12. Trzy konfiguracje robota pokazane w przestrzeni roboczej i przestrzeni konfiguracyjnej



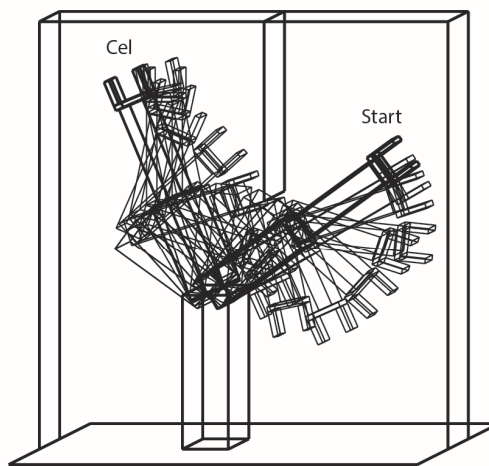
RYSUNEK 26.13. Graf widoczności. Linie łączą każdą parę wierzchołków „widzących się” nawzajem — linie, które nie przechodzą przez przeszkodę. Najkrótsza ścieżka musi zawierać te linie



**RYSUNEK 26.14.** Diagram Woronoja przedstawiający zbiór punktów (czarne linie) jednakowo odległe od dwóch (lub więcej) przeszkód w przestrzeni konfiguracyjnej

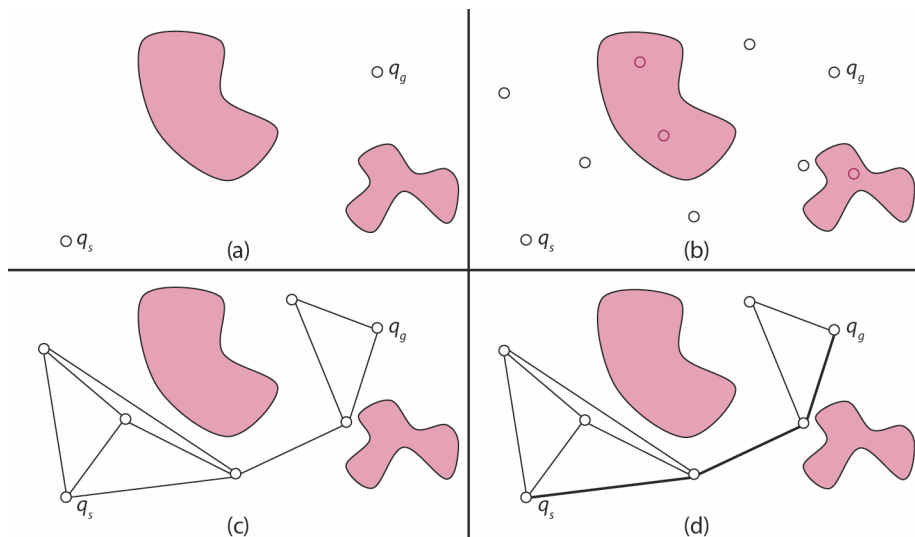


(a)

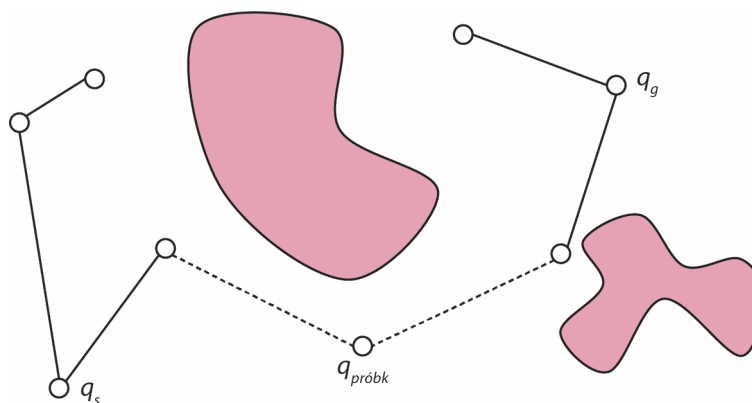


(b)

**RYSUNEK 26.15.** (a) Funkcja wartościująca i ścieżka znalezione dla komórek dyskretnego gridu aproksymującego ciągłą przestrzeń konfiguracyjną. (b) Ta sama ścieżka ukazana we współrzędnych przestrzeni roboczej. Zauważ, jak robot zgina łokieć, by uniknąć kolizji z pionową przeszkodą



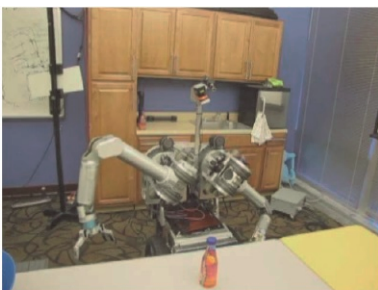
**RYSUNEK 26.16.** Algorytm probabilistycznej mapy drogowej (PRM). (a): Konfiguracja startu i celu. (b) Próbką  $M$  bezkolejnych kamieni milowych (tutaj  $M = 5$ ). (c) Połącz każdy kamień milowy z jego  $k$  najbliższymi sąsiadami (tutaj  $k = 3$ ). (d) Znajdź na grafie wynikowym najkrótszą ścieżkę od startu do celu



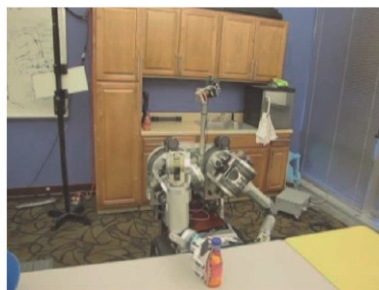
**RYSUNEK 26.17.** Dwukierunkowy algorytm RRT konstruuje dwa drzewa (jedno poczynawszy od startu, drugie poczynawszy od celu) poprzez przyrostowe łączenie każdej próbki z najbliższym węzłem w każdym z drzew, o ile takie połączenie jest możliwe. Gdy dana próbka łączy się z obydwoma drzewami, oznacza to, że znaleźliśmy żadaną ścieżkę



(a)

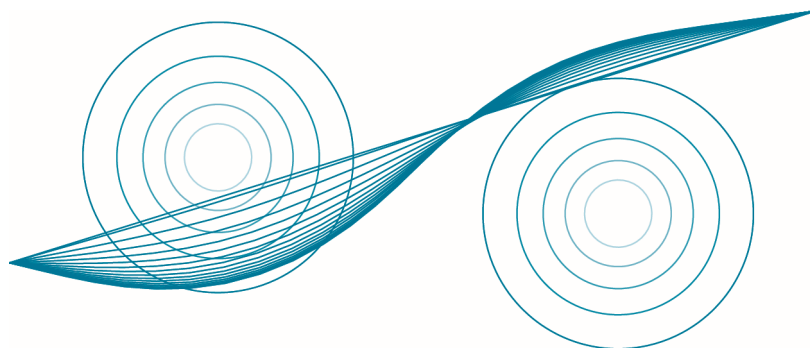


(b)

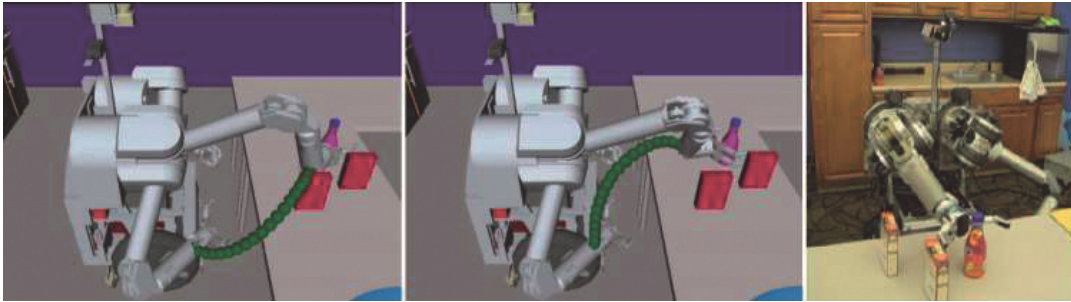


(c)

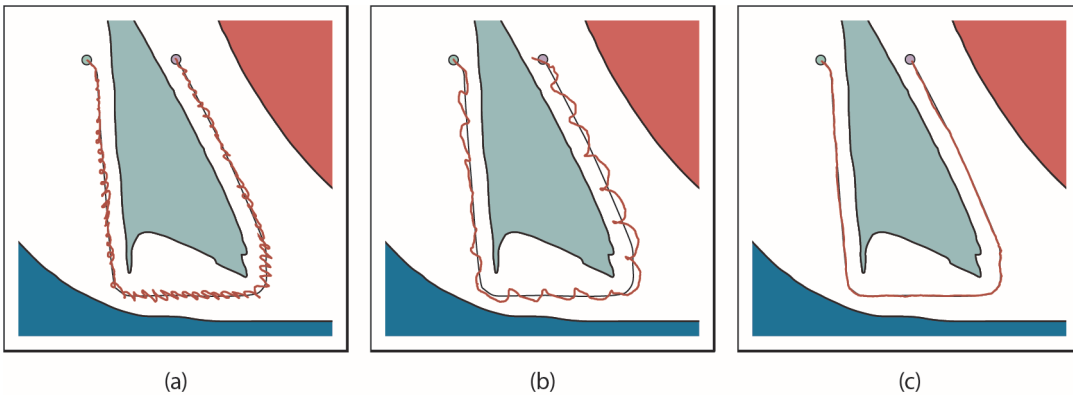
**RYSUNEK 26.18.** Migawki trajektorii wytworzone przez RRT i poddane obróbce końcowej ze skrótami. Dzięki uprzejmości: Anca Dragan



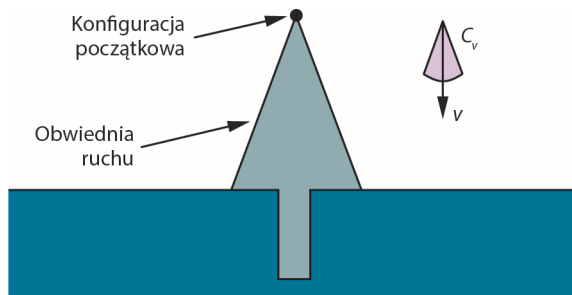
**RYSUNEK 26.19.** Optymalizacja trajektorii dla planowania ruchu. Dwie przeszkody punktowe wraz z otaczającymi je kolistymi pasami o malejącym koszcie. Optymalizator zaczyna od trajektorii prostoliniowej i wygina ją, omijając przeszkody i znajdując ścieżkę o minimalnym koszcie



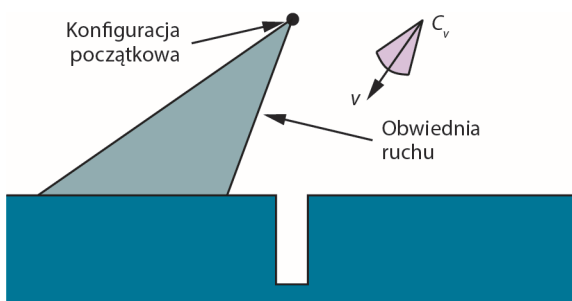
**RYSUNEK 26.20.** Zadanie dosięgnięcia butelki chwytakiem rozwiązywane za pomocą optymalizatora trajektorii. Po lewej: trajektoria początkowa, wykreślona dla efektora końcowego. W środku: wynikowa trajektoria po optymalizacji. Po prawej: konfigurowanie celu. Dzięki uprzejmości: Anca Dragan. Patrz (Ratliff i in, 2009)



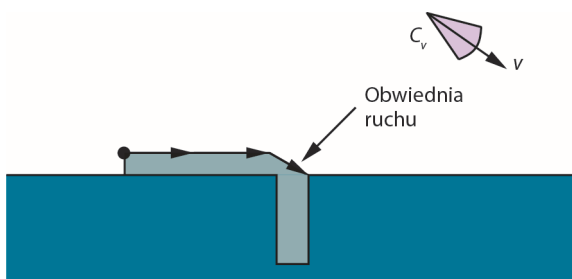
**RYSUNEK 26.21.** Sterowanie ramieniem robota za pomocą



**RYСУNEK 26.22.** Dwuwymiarowe środowisko, stożek niepewności prędkości i obwiednia możliwych ruchów robota. Zamierzona prędkość wynosi  $v$ , ale z powodu niepewności rzeczywista prędkość może zawierać się w dowolnym miejscu stożka  $C_v$ , co skutkuje ostateczną konfiguracją gdzieś w obwiedni ruchu. Oznacza to, że nie wiedzielibyśmy, czy trafiliśmy w otwór, czy nie

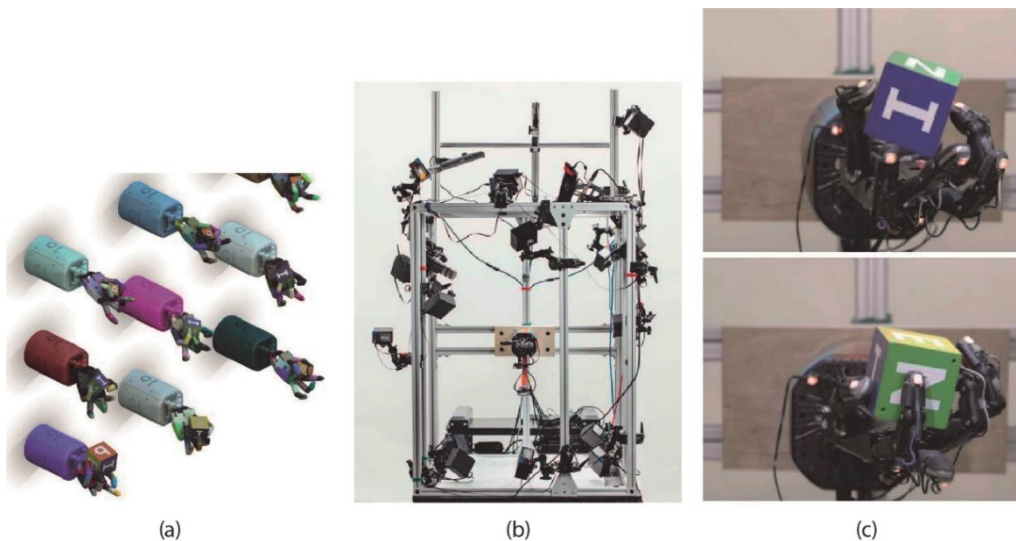


**RYСУNEK 26.23.** Pierwsze polecenie ruchu i wynikowa obwiednia możliwych ruchów robota. Bez względu na rzeczywisty ruch, wiemy, że ostateczna konfiguracja uplasuje się po lewej stronie otworu

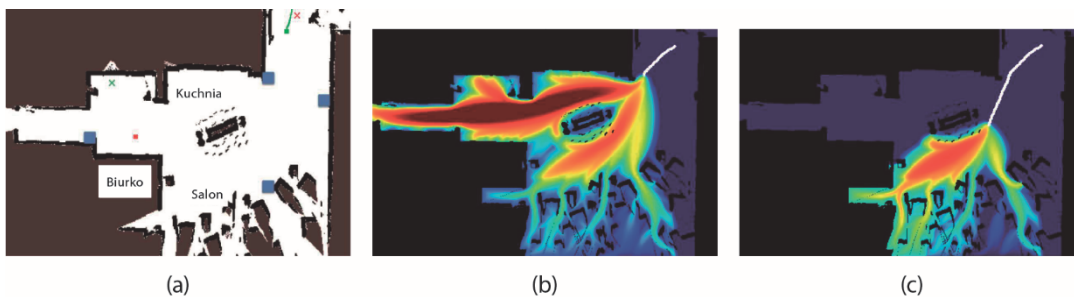


**RYСУNEK 26.24.** Drugie polecenie ruchu i obwiednia możliwych ruchów. Nawet mimo błędu, ostatecznie dostaniemy się do otworu

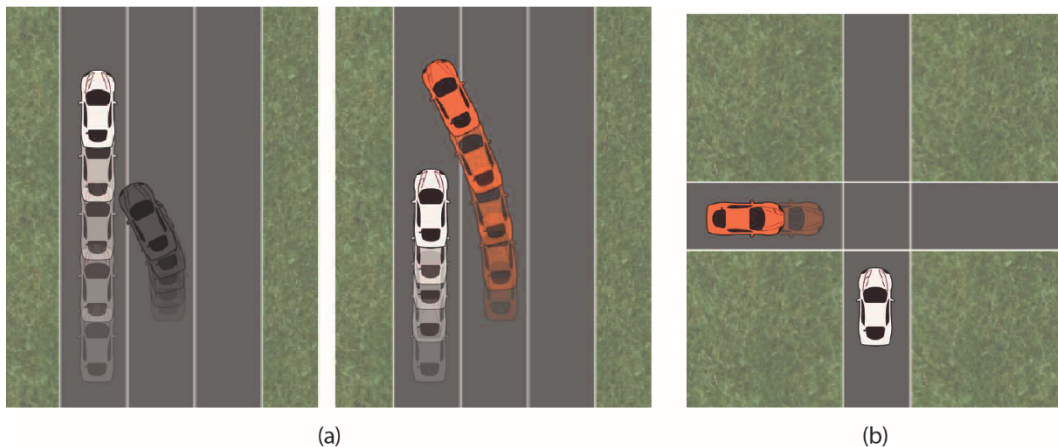




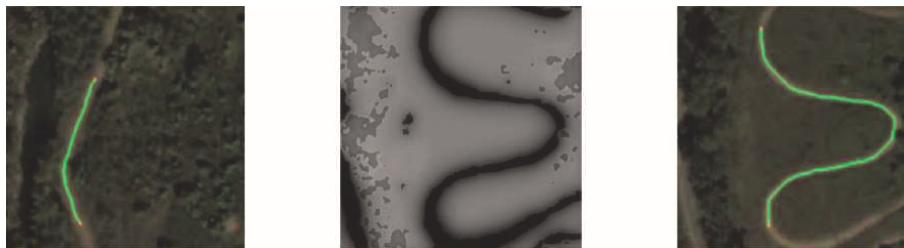
**RYSUNEK 26.25.** Trening w celu wyuczenia solidnej polityki. (a) Przeprowadzane są liczne symulacje manipulowania obiektami, z różnymi losowymi parametrami fizyki i oświetlenia. Dzięki uprzejmości Wojciecha Zaremby. (b) Rzeczywiste środowisko, z pojedynczym manipulatorem pośrodku klatki, otoczonej kamerami i dalmierzami. (c) Symulacja i trening w świecie rzeczywistym, dające wiele różnych polityk chwytania obiektów; tutaj chwyt szczypcowy i chwyt czworonożny. Dzięki uprzejmości OpenAI. Patrz Andrychowicz i in. (2018a)



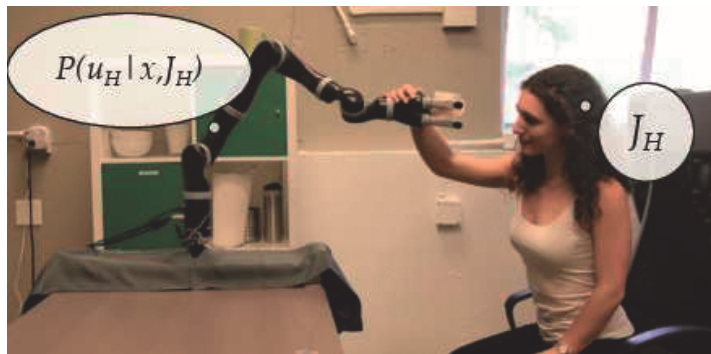
**RYSUNEK 26.26.** Dokonywanie przewidywań przy założeniu, że ludzie są bezwzględnie racjonalni w dążeniu do swych celów: robot wykorzystuje przeszłe akcje do aktualizowania swych przekonań na temat celu, do którego zmierza dana osoba, a następnie wykorzystuje te przekonania do przewidywania jej przyszłych akcji. (a) Mapa pomieszczenia. (b) Przewidywania robota po obejrzeniu niewielkiej części trajektorii osoby (biała ścieżka). (c) Przewidywania robota po obejrzeniu większej liczby ludzkich akcji: robot wie teraz, że dana osoba nie kieruje się do korytarza po lewej stronie, ponieważ w świetle dotychczasowej ścieżki jest to bardzo mało prawdopodobne. Zdjęcia dzięki uprzejmości Briana D. Ziebart. Patrz Ziebart i in., 2009



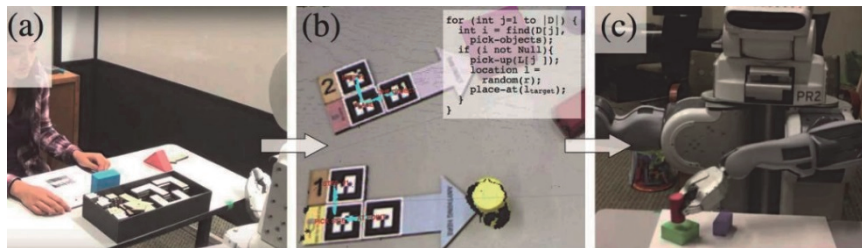
**RYСУNEK 26.27.** (a) **Po lewej:** autonomiczny samochód (na środkowym pasie) przewiduje, że kierowca na lewym pasie nie zamierza go przepuścić, więc sam zwalnia, by przejechać na lewy pas za wspomnianym kierowcą. **Po prawej:** autonomiczny samochód bierze pod uwagę fakt, że jego akcje mają wpływ na akcje kierowców-ludzi, więc przyspiesza zakładając, że kierowca jadący lewym pasem przepuści go. (b) Ten sam algorytm na skrzyżowaniu wyzwała niezwykłą strategię: samochód autonomiczny cofa się nieznacznie, by w ten sposób zwiększyć prawdopodobieństwo, że kierowca-człowiek przejedzie przez skrzyżowanie jako pierwszy. Zdjęcia dzięki uprzejmości: Anca Dragan. Zobacz Sadigh i in. (2016)



**RYСУNEK 26.28.** **Po lewej:** mobilny robot obejrzał demonstrację, z której wynika, że lepiej pozostać na polnej drodze niż zapuszczać się na teren trawiasty. **Pośrodku:** robot wywnioskowuje żądaną funkcję kosztu i wykorzystuje ją w nowej scenie, wiedząc, jak obniżyć koszty na drodze. **Po prawej:** robot planuje ścieżkę dla nowej sceny, która to ścieżka poprowadzi przez polną drogę zgodnie z preferencjami poznanymi w trakcie demonstracji. Zdjęcia dzięki uprzejmości Nathana Ratliffa i Jamesa A. Bagnella. Patrz Ratliff i in. (2006)



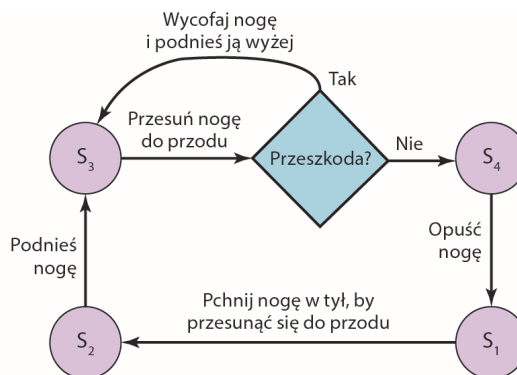
**RYSUNEK 26.29.** Nauczycielka popycha ramię robota w dół, ucząc go w ten sposób pozostawania jak najbliżej stołu. Robot odpowiednio aktualizuje swoje rozumienie pożądanej funkcji kosztu i zaczyna ją optymalizować. Zdjęcie dzięki uprzejmości: Anca Dragan. Patrz Bajcsy i in. (2017)



**RYSUNEK 26.30.** Interfejs programistyczny w postaci specjalnie zaprojektowanych bloków w przestrzeni roboczej robota; programowanie akcji przebiega na wysokim poziomie — odbywa się przez wybieranie tychże bloków. Zdjęcia dzięki uprzejmości Mayi Cakmak. Patrz Sefidgar i in. (2017)



(a)



(b)

**RYSUNEK 26.31.** (a) Czyngis, sześcionożny robot. (Zdjęcie dzięki uprzejmości Rodney’a A. Brooksa). (b) Augmentowany automat skończony (AFSM), który steruje jedną nogą robota. Automat ten reaguje na sprzężenie zwrotne czujnika: jeśli noga utknie podczas próby przekroczenia przeszkody, robot będzie ją podnosił coraz wyżej



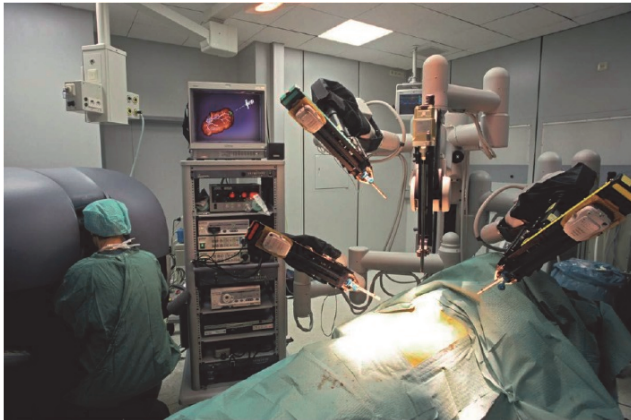
(a)



(b)

**RYSUNEK 26.32.** (a) Pacjent z interfejsem mózg-maszyna sterującym ramieniem robota chwytającego butelkę z napojem. Zdjęcie dzięki uprzejmości Brown University. (b) Roomba — robot-odkurzacz. Zdjęcie: HANDOUT/KRT/Newscom





(a)

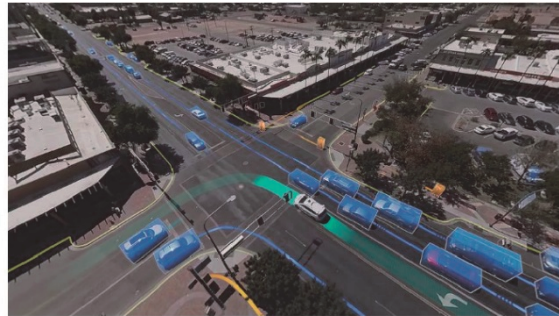


(b)

**RYSUNEK 26.33.** (a) Robot chirurgiczny na sali operacyjnej. Zdjęcie: Patrick Landmann/Science Source. (b) Robot dostarczający akcesoria medyczne w szpitalu. Zdjęcie: „Wired”



(a)

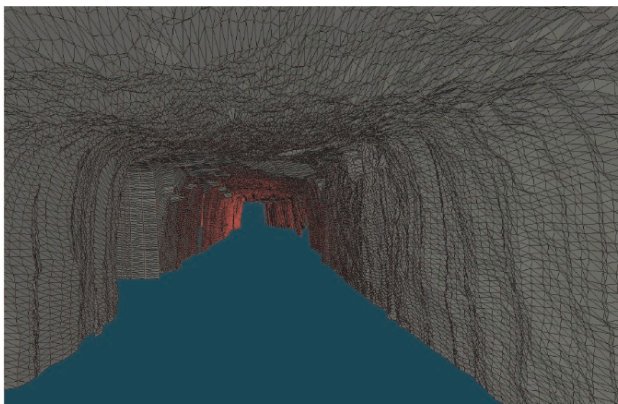


(b)

**RYSUNEK 26.34.** (a) Autonomiczny samochód BOSS, który wygrał wyścig DARPA Urban Challenge. Zdjęcie: Tangi Quemener/AFP/Getty Images/Newscom. Dzięki uprzejmości Sebastiana Thruna. (b) Widok z lotu ptaka, pokazujący percepcję i przewidywania autonomicznego samochodu Waymo (biały pojazd z zaznaczoną zieloną trasą). Pokazane są przewidywane trajektorie innych pojazdów (niebieskie pola) i pieszych (pomarańczowe pola). Granice drogi/chodnika zaznaczone są na żółto. Zdjęcie dzięki uprzejmości Waymo



(a)



(b)

**RYSUNEK 26.35.** (a) Robot eksplorujący opuszczoną kopalnię węgla. (b) Mapa 3D kopalni sporządzona przez tegoż robota. Dzięki uprzejmości Sebastiana Thruna

---