Roboty Manipulacyjne i Mobilne
Budowanie mapy, lokalizacja, SLAM

Janusz Jakubiak

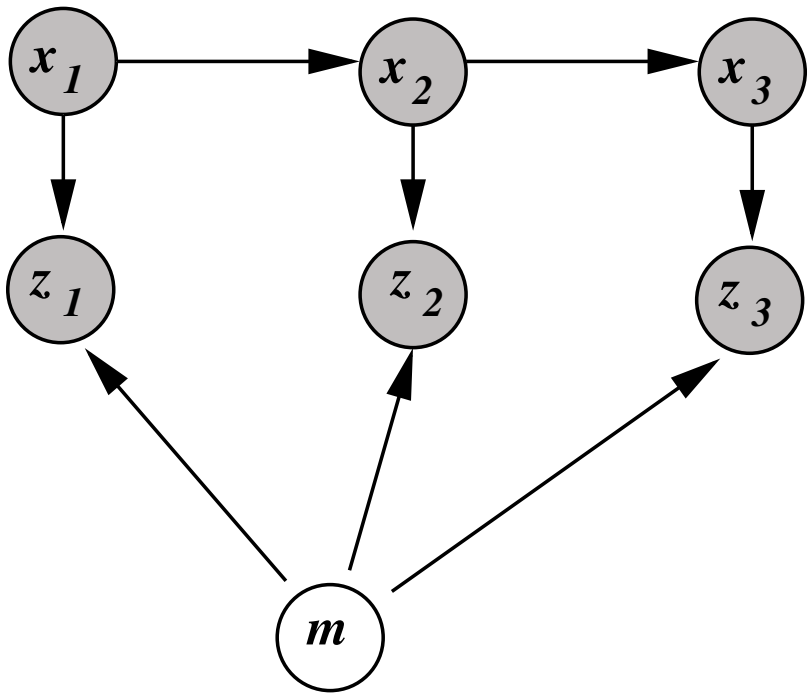
Zakład Podstaw Cybernetyki i Robotyki
Instytut Informatyki, Automatyki i Robotyki Politechnika Wrocławska

4.05.2012

Budowanie mapy

Wersja I zadania (prostsza)

Na podstawie **znanego** ciągu pozycji robota (*trajektorii*) oraz odczytów sensorów (*obserwacji*) w tych pozycjach – wyznaczyć położenia przeszkód i obszary wolne w otoczeniu robota (mapę *w wybranej reprezentacji*).



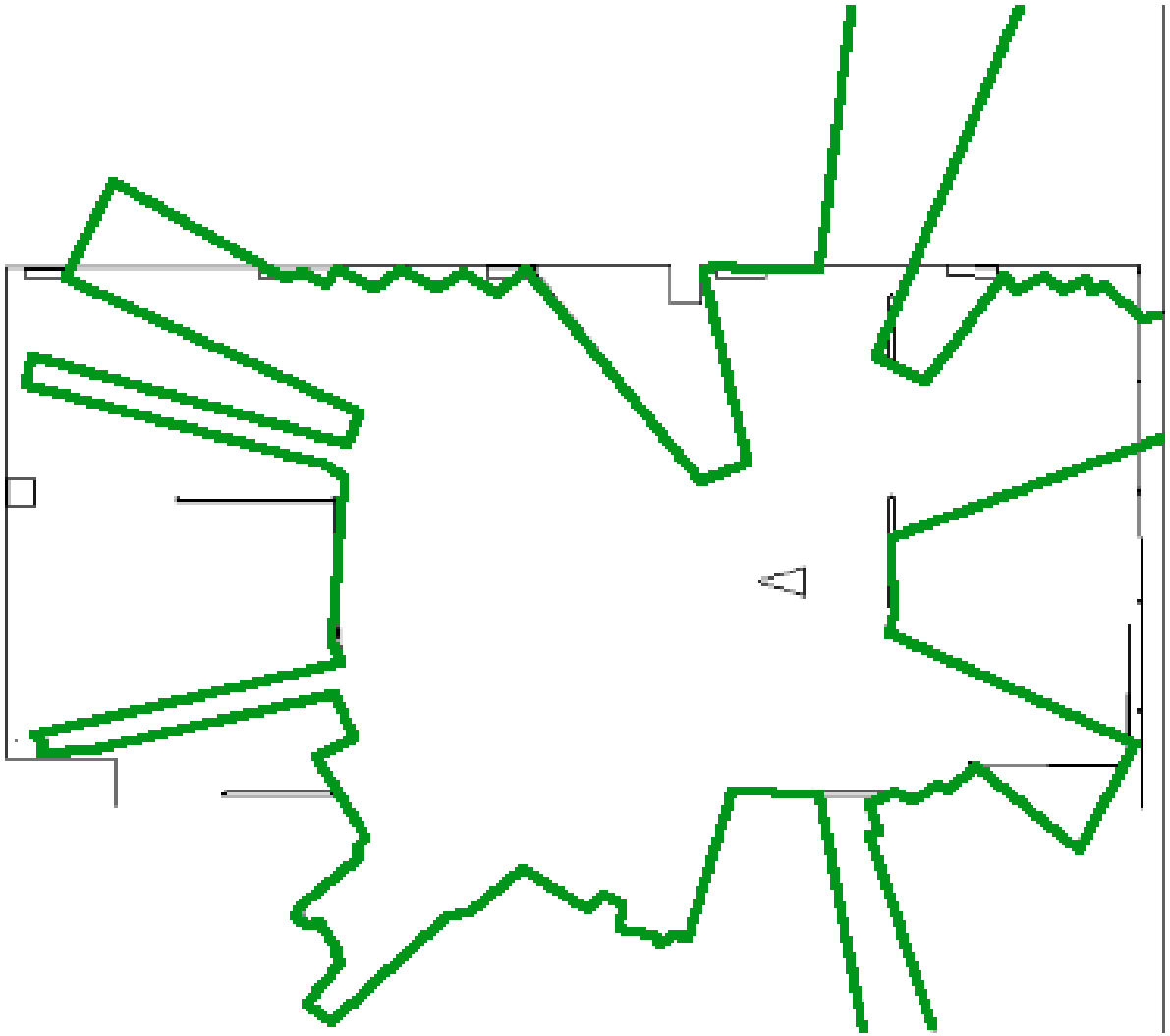
- m – mapa
- x_i – pozycja robota w kroku i
- z_i – obserwacja w kroku i

Budowanie mapy – sensory

Do zadania tworzenia mapy otoczenia wykorzystywane są sensory

- sonary,
- dalmierze laserowe,
- systemy mono- i stereowizyjne
- informacje o pozycji i orientacji robota z podsystemu samolokalizacji

Sonar jako skaner



[0]Leonard, Durrant-Whyte. Directed Sonar Sensing for Mobile Robot Navigation

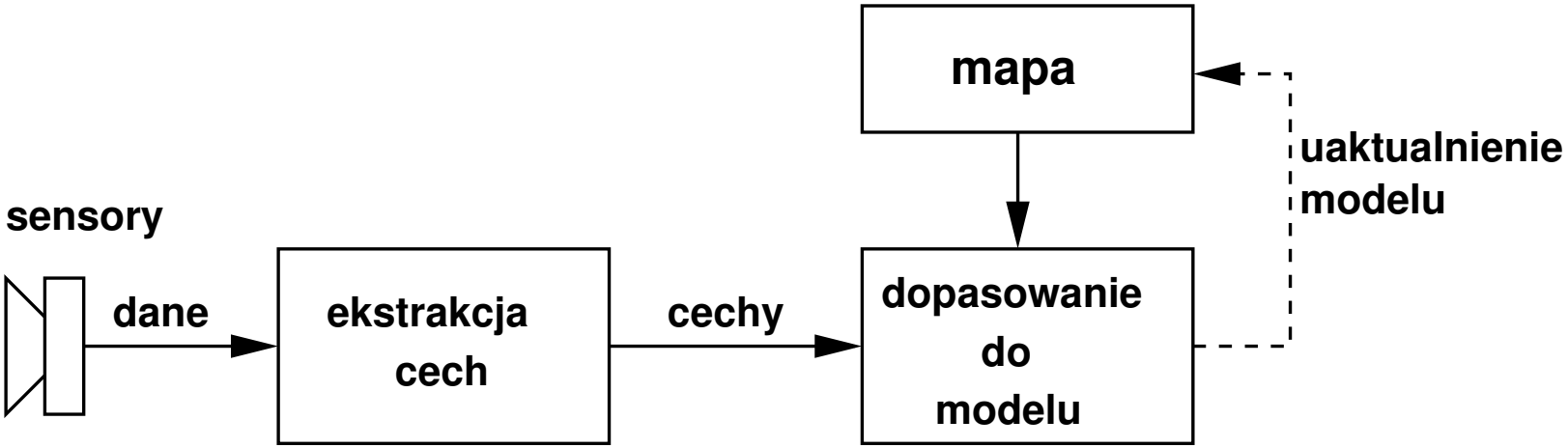
Budowanie mapy – cechy

Wyszukiwane cechy

- linie (odcinki)
- narożniki
- znaczniki

Wykryte cechy(elementy) nawet w statycznym środowisku powinny być śledzone podczas ruchu robota celem weryfikacji i ewentualnej korekty mapy.

Budowanie mapy – schemat



Etapy tworzenia mapy [Crowley 1989]

1. Tworzenie opisu wyższego poziomu na podstawie bieżących odczytów czujników i modeli sensorów.
2. Dopasowanie cech rozpoznanych przez sensory do bieżącego opisu otoczenia robota.
3. Aktualizacja opisu otoczenia poprzez osłabienie lub wzmocnienie prawdopodobieństw obecności obiektów.

Podział map

W zależności od zmienności

- statyczne
- dynamiczne

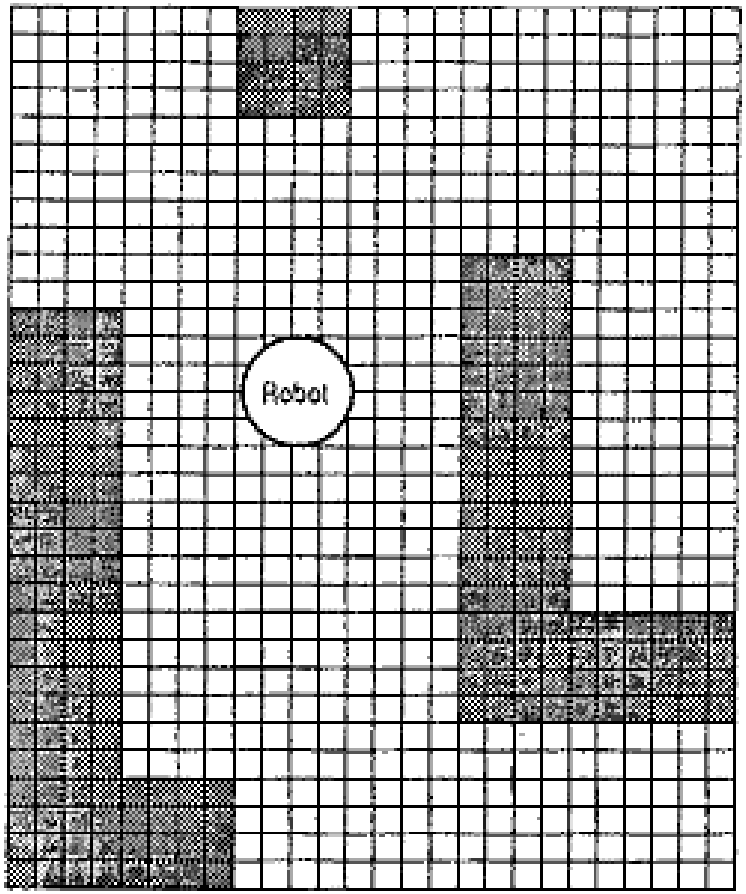
W zależności od rodzaju opisu

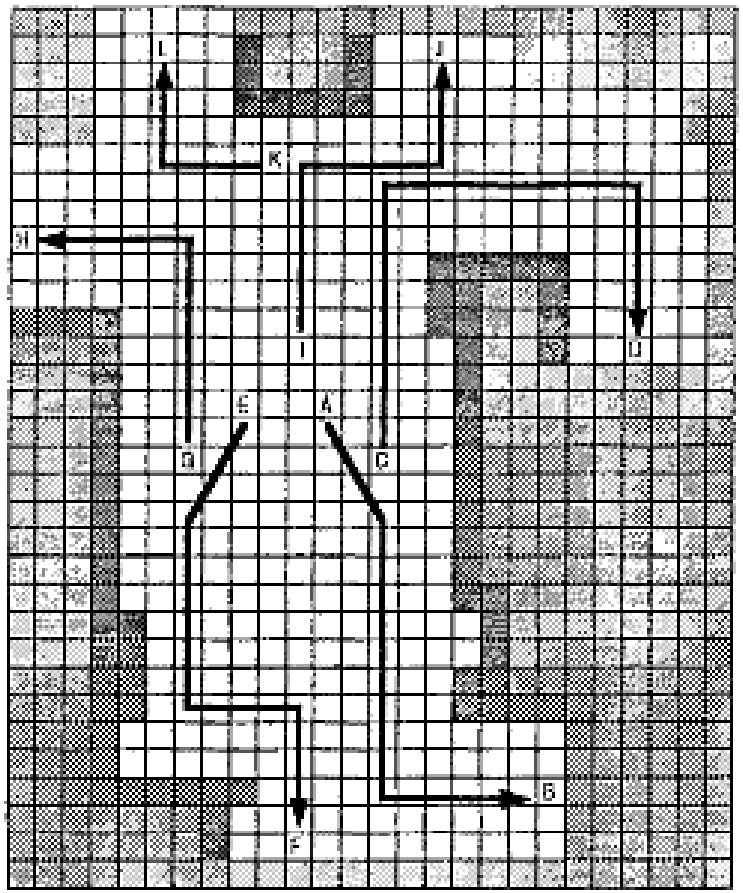
- metryczne
- topologiczne
- hybrydowe

Według opisu przestrzeni

- rastrowe
- wektorowe

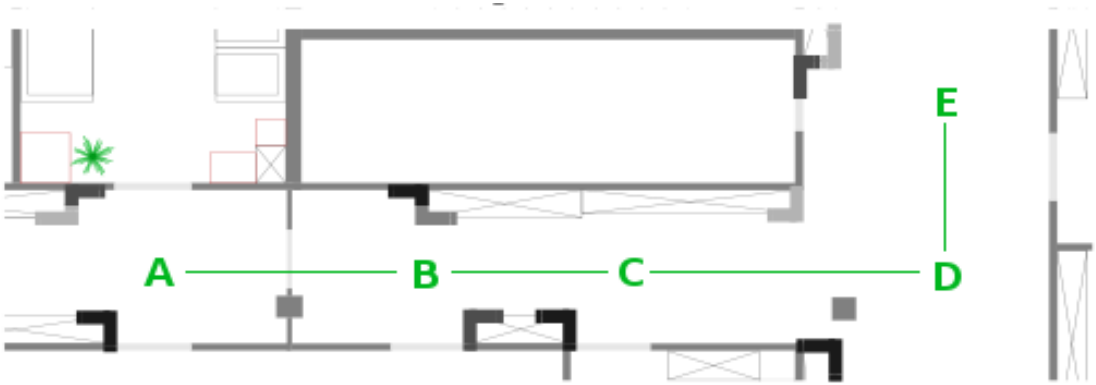
Mapa rastrowa, metryczna

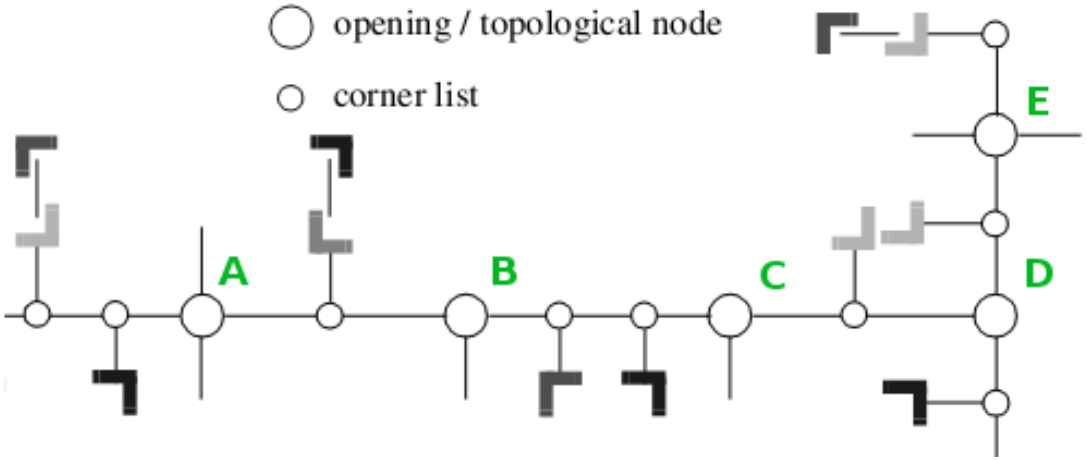




[0] A. Zelinsky. Environment Mapping with a Mobile Robot using Sonar

Mapa topologiczna





[0]Tomatis i in. Combining topological and metric: a natural integration for SLAM

Porównanie map metrycznych i topologicznych

Zalety map metrycznych

- łatwość tworzenia, reprezentacji, modyfikacji
- rozpoznawanie miejsc niezależne od położenia/orientacji
- umożliwiają generowanie optymalnych (najkrótszych) ścieżek

Zalety map topologicznych

- złożoność mapy zależy od złożoności obszaru, nie jego wielkości
- nie wymagają znajomości dokładnej pozycji robota
- reprezentacja łatwiejsza do przetwarzania wysokopoziomowego

Problem

Dlaczego tworzenie map jest problemem trudnym?

Przestrzeń możliwych map jest bardzo duża: w przypadku ciągłym – nieskończeniewymiarowa, w dyskretnym – często rzędu $10^4 - 10^6$.

Główne czynniki określające złożoność:

- rozmiar – wielkość środowiska względem zasięgu sensorów,
- zakłócenia pomiarów i sterowania,
- niejednoznaczność percepcji (możliwość dopasowania różnych punktów otoczenia do tego samego punktu mapy),
- cykle (możliwość dotarcia do tego samego punktu różnymi drogami, przy jednoczesnych błędach określania pozycji).

Praktyczne ograniczenia

- wielkości fizyczne możliwe do zmierzenia
- zasięg sensorów
- zakłócenia
- niejednoznaczność interpretacji
- niedokładność wyznaczania pozycji robota
- dynamika otoczenia
- wymagania pracy w czasie rzeczywistym

Rastrowa mapa zajętości (*occupancy grid*)

Szukamy mapy, która maksymalizowałaby prawdopodobieństwo

$$p(m|z_{1:t}, x_{1:t}).$$

Dzielimy mapę na komórki $m = \{m_i\}$, każda z prawdopodobieństwem zajętości (1 – zajęta, 0 – wolna).

Problem rozmiaru:

Mapa 100x100 to 10 000 komórek. Liczba możliwych map: 2^{10000}.

Rozwiązanie standardowe:

Obliczamy prawdopodobieństwo każdej komórki oddzielnie

$$p(m|z_{1:t}, x_{1:t}) \simeq \Pi_i \; p(m_i|z_{1:t}, x_{1:t})$$

Rastrowa mapa zajętości (2)

Do aktualizacji prawdopodobieństwa korzystamy z binarnego filtra bayesowskiego i reprezentacji logarytmicznej

$$l_t = l_{t-1} + \log \frac{p(x|z_t)}{1 - p(x|z_t)} - \log \frac{p(x)}{1 - p(x)}$$

Rastrowa mapa zajętości – algorytm

$l_0 = \log \frac{p(m_i=1)}{p(m_i=0)} = \log \frac{p(m_i)}{1-p(m_i)}$

foreach m_i do

  if m_i in perception_field(z_t)

    $l_{t,i} = l_{t-1,i} + inverse\_sensor\_model(m_i, x_t, z_t) - l_0$

  else

    $l_{t,i} = l_{t-1,i}$

  endif

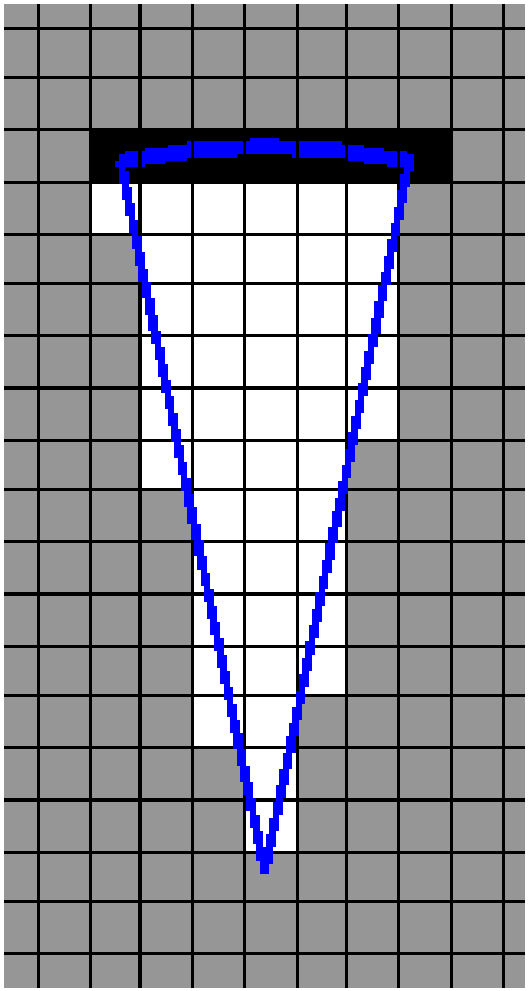
endfor

$p(m_i|z_{1:t}, x_{1:t}) = 1 - \frac{1}{1+\exp\{l_{t,i}\}}$

$inverse\_sensor\_model(m_i, x_t, z_t) = \log \frac{p(m_i|z_t,x_t)}{1-p(m_i|z_t,x_t)}$

Model czujnika

Wersja prosta:



Wersja zaawansowana:

Niech
(r_i, φ_i) – odległość i kąt do komórki m_i,
(θ, β) – orientacja i szerokość wiązki sensora
α – grubość przeszkody

if $r_i > \min(z_{max}, z_t + \alpha/2)$ or $|\varphi_i - \theta_{sens}| > \beta/2$
    return l_0
if $z_t < z_{max}$ and $|r_i - z_t| < \alpha/2$
    return l_{zaj}
if $r_i \leqslant z_t$
    return l_{wol}

Mapy zajętości a fuzja sensorów

Pytanie

W jaki sposób stworzyć mapę zajętości korzystając z różnych sensorów?

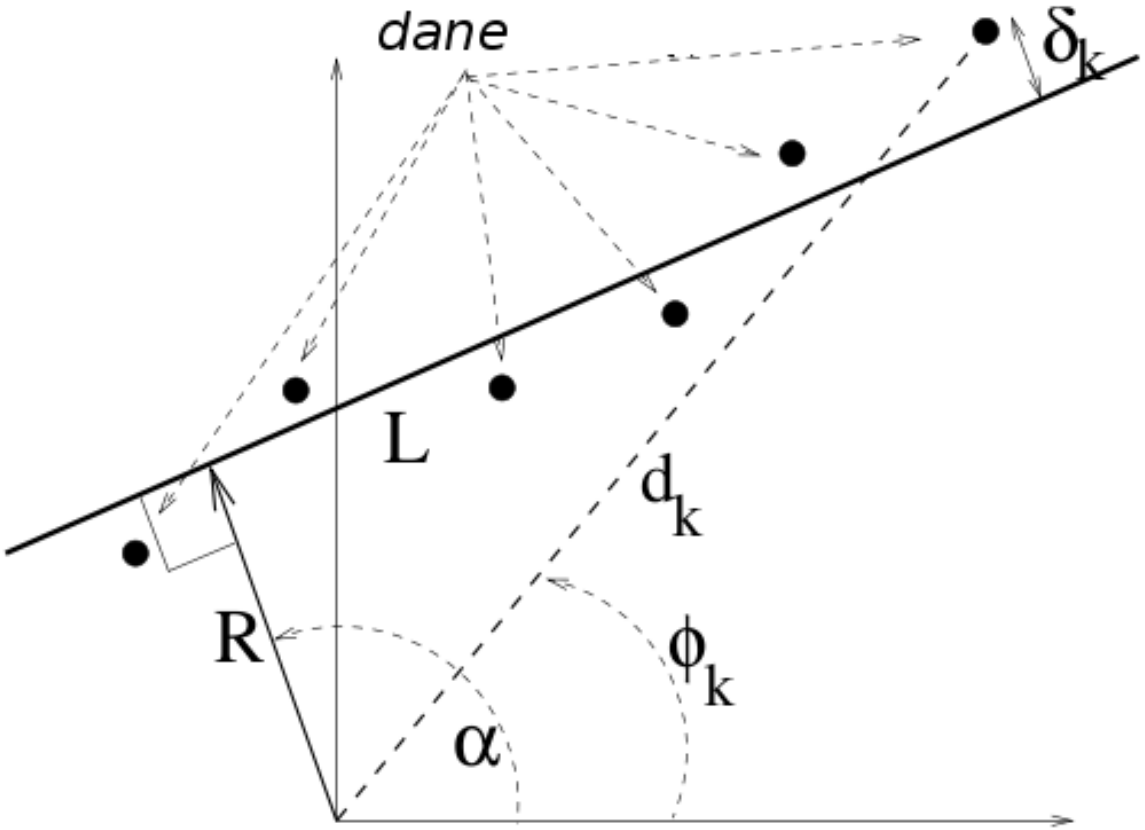
1. (sposób prosty) Uruchomić algorytm dla tej samej mapy kolejno dla każdego z czujników. Wada: Jeśli różne typy czujników wykrywają różne przeszkody, mapa będzie niestabilna.
2. (podejście typowe) Stworzyć oddzielne mapy dla każdego z typów czujników i połączyć je odpowiednią funkcją, np. jeśli pomiary są niezależne, korzystając z praw De Morgana:

$$p(m_i) = 1 - \Pi_k(1 - p(m_i^k)),$$

lub wg. najbardziej pesymistycznej hipotezy

$$p(m_i) = \max_k p(m_i^k).$$

Mapy wektorowe – wykrywanie linii



$$\delta_k = d_k \cos(\alpha - \phi_k) - R$$

[0]por. transformata Hough'a

[0]Pfister i in. Weighted line fitting algorithms...

Wykrywanie linii

Niech

- $d_k = D_k + \varepsilon_d$, z ε_d będącym błedem odczytu sensorów odległości o rozkładzie normalnym z wariancją σ_d^2,
- $\phi_k = \Phi_k + \varepsilon_\phi$ – analogicznie błędem orientacji o wariancji σ_ϕ^2.

Błąd dopasowania do linii k-tego pomiaru

$$P_k = E\{\delta_k \delta_k^T\} = \sigma_d^2 \cos^2(\alpha - \Phi_k) + \sigma_\phi^2 D_k^2 \sin^2(\alpha - \phi_k)$$

Estymata R

$$\hat{R} = P_{RR} \left(\sum_k \frac{d_k \cos(\hat{\alpha} - \phi_k)}{P_k} \right), \qquad P_{RR} = \left(\sum_k P_k^{-1} \right)$$

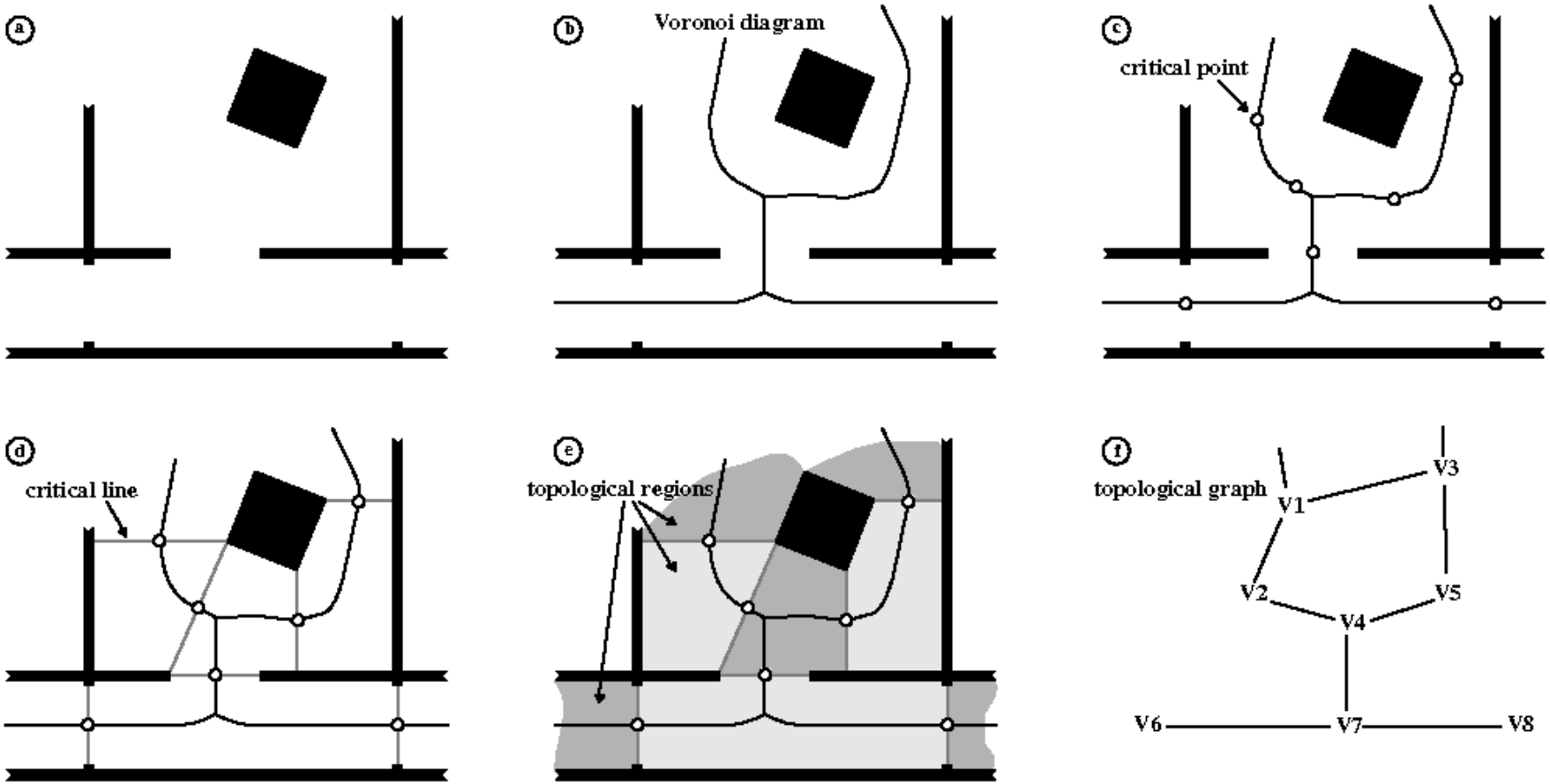
Estymatę α wyznacza się np. przez numeryczną maksymalizację prawdopodobieństwa a posteriori.

Wykrywanie linii – RANSAC

Algorytm RANdom SAmpling Consensus dla dalmierza laserowego

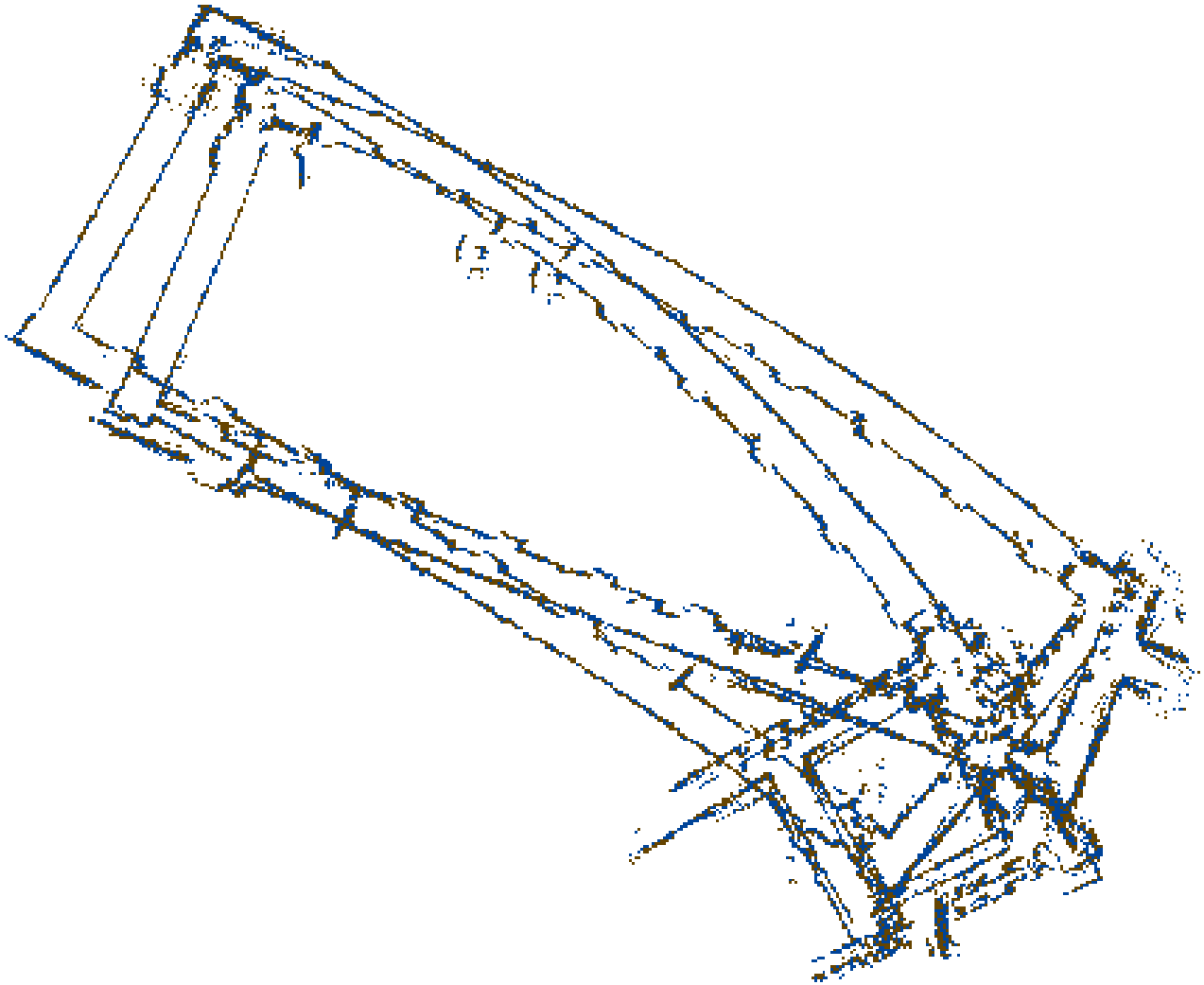
- **dopóki** istnieją próbki nie przypisane do żadnej linii, a liczba prób jest mniejsza od N
  - wybierz dowolny odczyt r_k
  - wybierz losowo S próbek w zakresie do D stopni od r_k (R_s)
  - dla $\{r_k\} \cup R_s$ dopasuj linię metodą najmniejszych kwadratów
  - wyznacz ile z wszystkich próbek leży w odległości powyżej X od tej linii
  - **jeśli** liczba odczytów pasujących do linii jest większa od C
    - dla punktów pasujących do linii wyznacz ponownie parametry prostej
    - dodaj wyznaczoną linię do zbioru wykrytych linii
    - usuń uwzględnione punkty ze zbioru punktów do przetworzenia

Konwersja mapy metrycznej do topologicznej



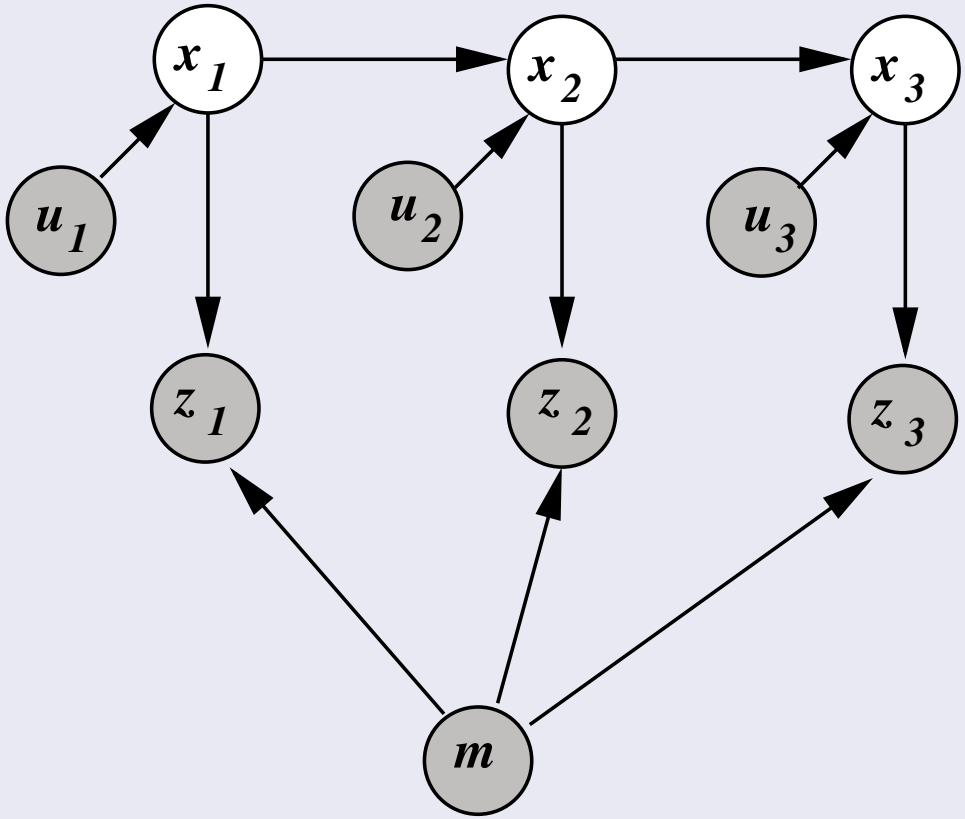
[0]Thrun. Learning maps for indoor mobile robot navigation

Wpływ błedów odometrii – krok do SLAM



[0]Thrun. Integrating topological and metric maps...

Zadanie lokalizacji



Na podstawie znanych: modelu otoczenia m, ciągu sterowań $\mathcal{U} = \{u_{1:t}\}$ i obserwacji $\mathcal{Z} = \{z_{1:t}\}$ wyznaczyć stan robota x_t (lub ciąg stanów $\mathcal{X} = \{x_{1:t}\}$).

Zadanie lokalizacji – systematyka

Informacja początkowa

- śledzenie pozycji (lokalizacja lokalna)
- lokalizacja globalna
- problem „porwanego robota”

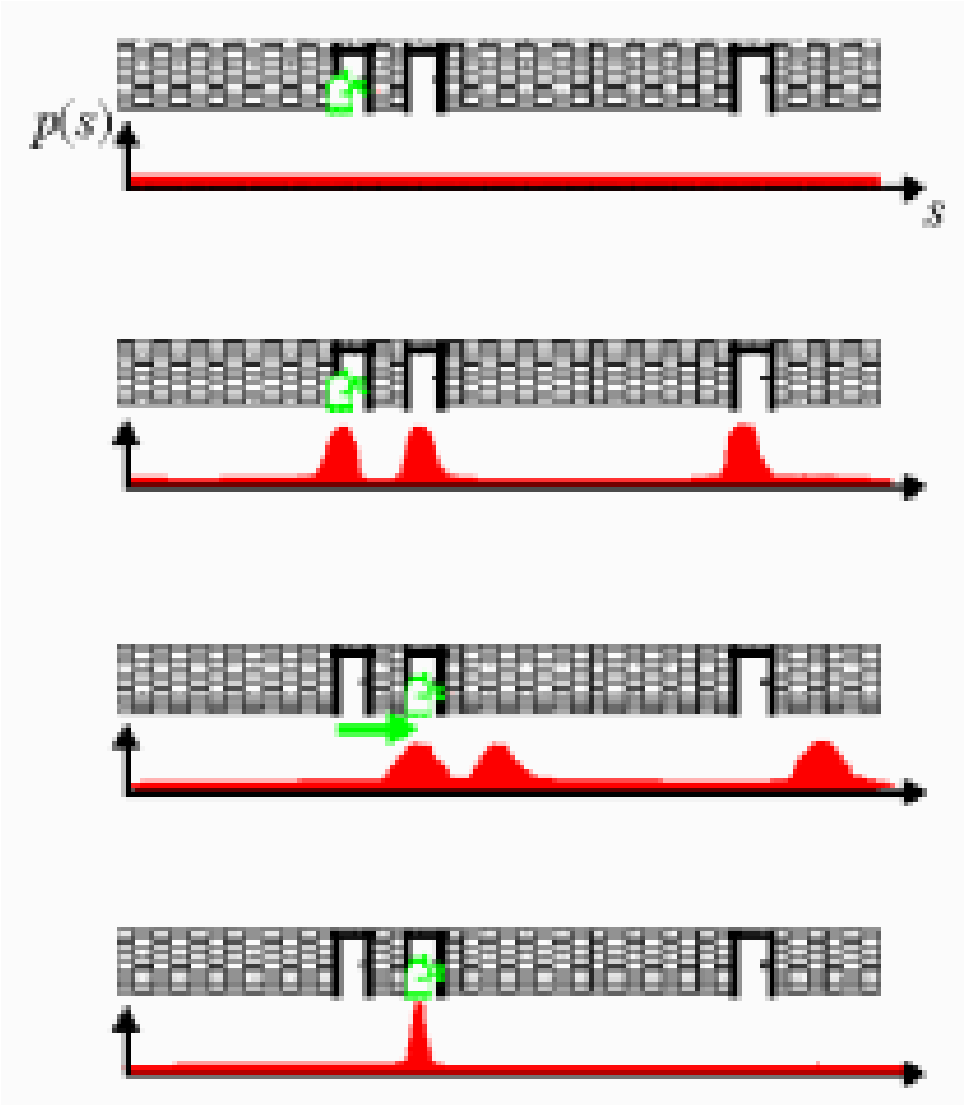
Środowisko

- statyczne
- dynamiczne

Wpływ na sterowanie

- bierna
- aktywna

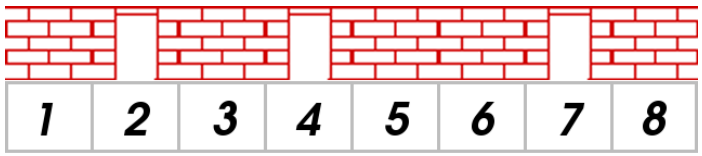
Zadanie lokalizacji – idea



[0]http://www.cs.cmu.edu/ thrun/tutorial/sld027.htm

Zadanie lokalizacji – przykład

$$p(z_t = \text{  } \,|x_t = x_i) = \begin{cases} 1 & \text{dla } x_i = 2, 4, 7, \\ 0 & \text{dla } x_i = 1, 3, 5, 6, 8 \end{cases}$$



$$x_{t+1} = f(x_t, u_t = 1) = x_t + 1$$

$$Z = \left\{ \text{  }, \text{  }, \text{  } \right\}$$

$p(x_t = x_i | z_{1:t})$:

i, $z_i \backslash x_i$	1	2	3	4	5	6	7	8
0	1/8	1/8	1/8	1/8	1/8	1/8	1/8	1/8
1 	0	1/3	0	1/3	0	0	1/3	0
2 	0	0	1/3	0	1/3	0	0	1/3
3 	0	0	0	1	0	0	0	0

Zadanie lokalizacji – przykład, c.d.

Uwaga

Prawdopodobieństwo $p(z_t|x_t)$ zwykle nie jest binarne, także zmiana stanu może być zakłócona.

Pytanie

Jak wyglądałoby rozwiązanie, gdyby przyjąć, że prawdopodobieństwo, że drzwi są otwarte wynosiło $1/2$, a robot nie odróżniał zamkniętych drzwi od ściany? Gdyby dodatkowo $u_i = 0.9$, czyli zmiana stanu opisana byłaby równaniem

$$p(x_{t+1} = x_i + 1|x_t = x_i) = 0.9 \qquad p(x_{t+1} = x_i|x_t = x_i) = 0.1$$

(por. algorytm Viterbiego)

Lokalizacja wg. schematu Markowa

W każdej iteracji – dwie fazy:

- predykcja
  obliczenie przewidywanego rozkładu prawdopodobieństwa nowego stanu
  $$\bar{bel}(x_t = x) = \int p(x_t|x_{t-1}, u_t) bel(x_{t-1}) dx_{t-1}$$
- korekta
  hipotezy na podstawie odczytów sensorów z_t
  $$bel(x_t = x) = \eta p(z_t|x_t) \bar{bel}(x_t)$$

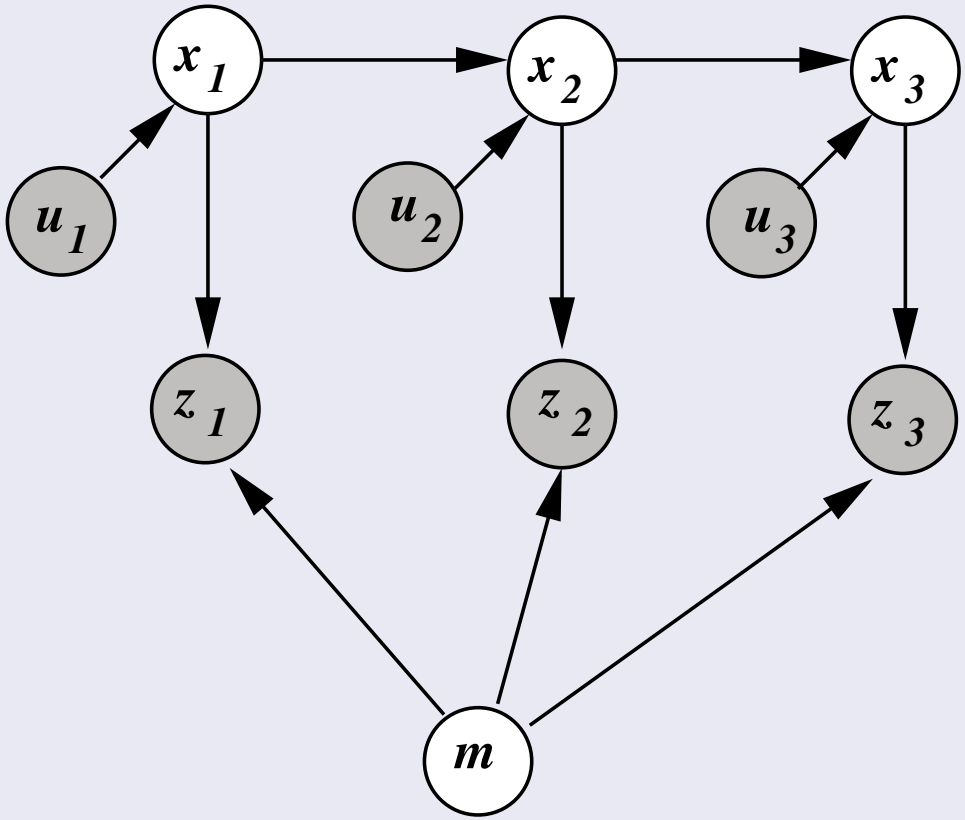
Krok do SLAM[1]

SLAM

Ponieważ zarówno wyznaczenie położenia robota (lokalizacja), jak i pomiary odległości do obiektów znajdujących się w otoczeniu robota (tworzenie mapy) obarczone są błędami, należy oba zadania rozpatrywać łącznie. Zadanie będzie (wielokotnie) bardziej złożone, jednak będzie można uzyskać dokładniej wyznaczone pozycję robota i przeszkód w jego otoczeniu.

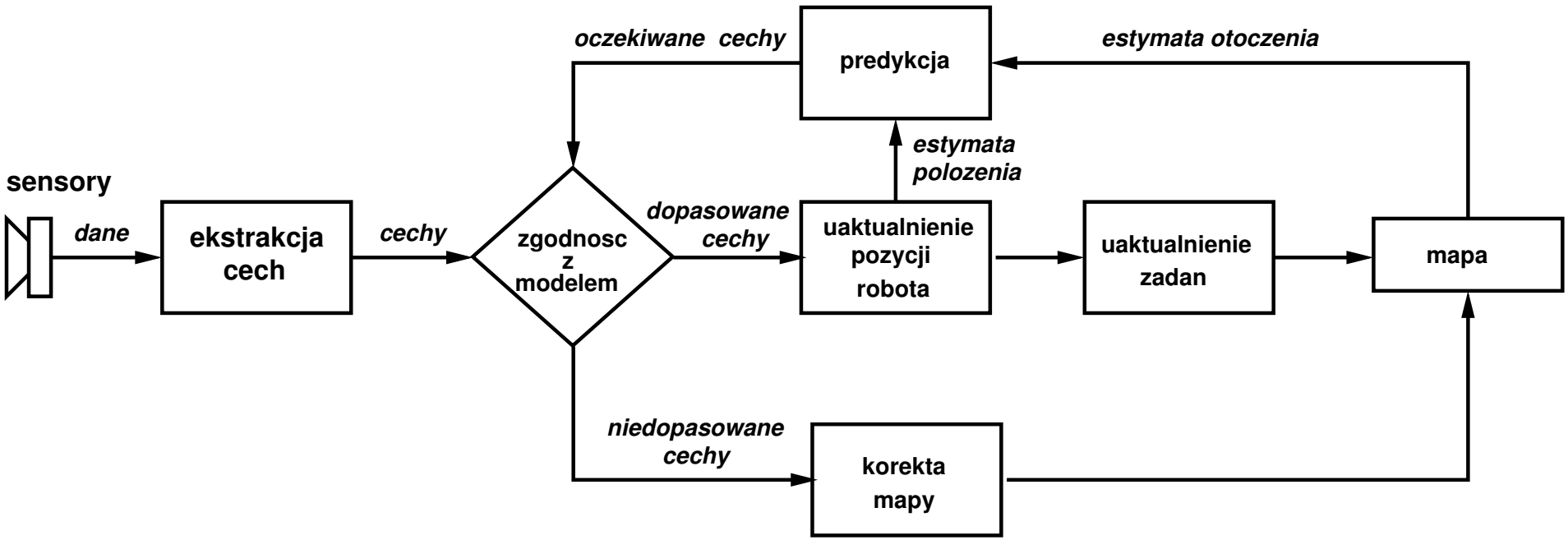
[1] SLAM (*Simultaneous localization and mapping*) – jednoczesna lokalizacja (robota) i budowa mapy

SLAM formalnie



Na podstawie znanych sterowań $\mathcal{U} = \{u_{1:t}\}$ i obserwacji $\mathcal{Z} = \{z_{1:t}\}$ wyznaczyć stany robota $\mathcal{X} = \{x_{1:t}\}$ i model otoczenia (mapę) m.

SLAM – schemat



SLAM – najważniejsze metody

- metody gaussowskie, np. rozszerzony filtr Kalmana (EKF)
- grafowe metody optymalizacji
- metody gridowe/Monte Carlo, np. filtry cząsteczkowe (*particle filter*)

EKF

- najstarsza z metod SLAM
- zakłada metryczną reprezentację świata z wyróżnionymi elementami stanowiącymi wektory w przestrzeni parametrów
- pozycja robota oraz wyróżnionych elementów świata tworzą sieć relacji obaczonych niepewnością

$$p(x_t, m|Z_T, U_T) \sim \mathcal{N}(\mu_t, \Sigma_t)$$

x_t– pozycja robota w chwili t, U_T – zbiór względnych przemieszczeń robota w czasie $[0, t]$, m – rzeczywista mapa otoczenia, Z_T – zbiór pomiarów otoczenia, μ_t – wektor stanu (poł. robota i elementów otoczenia), Σ_t – macierz kowariancji do oceny błędu przyjętego stanu μ_t

EKF, c.d.

Założenie

Mapa jest reprezentowana przez skończony zbiór cech. W wersji prostszej algorytmu obserwacje są jednoznacznie kojarzone z cechami, w innych wersjach – dopasowanie obserwacji do cech następuje np. przez estymację największego prawdopodobieństwa (*maximum likehood estimator, MLE*).

Własności

- jako reprezentacja gaussowska, nie uwzględnia sztywnych ograniczeń przestrzeni
- prosty EKF jest mało odporny na błędne dopasowania
- nie uwzględnia informacji negatywnych

Rozszerzenia: MHT, UKF, inne.

Grafowe metody optymalizacji

- przy każdym pomiarze dodawane są do grafu: węzeł z bieżącą pozycją i łuki pomiarów odległości do znaczników
- graf jest optymalizowany (analogicznie do minimalizacji układu mas-sprężyn)

$$X_t^\star, m^\star = \arg\max_{X_T, m} \log p(X_T, m | Z_T, U_T)$$

przy

$$\log p(X_T, m | Z_T, U_T) = \text{const} + \\ + \sum_t \log p(x_t | x_{t-1}, u_t) + \log p(z_t | x_t, u_t)$$

Metody gridowe/Monte Carlo

Własności

- Nie wymagają ekstrakcji cech
- Mogą uwzględniać informację negatywną
- Są nieparametryczne – nie są ograniczone założoną dystrybucją

Lokalizacja gridowa

$\{p_{k,0}\}$: początkowy rozkład prawdopodobieństw na siatce

foreach k do

$\bar{p}_{k,t} = \sum_i p_{t,t-1} \cdot motion\_model(x_{k\,sr}, u_t, x_{i\,sr})$

$p_{k,t} = \eta\, \bar{p}_{k,t} \cdot measurement\_model(z_t, x_{t\,sr}, m)$

endfor

- stosowany dyskretny filtr Bayesa
- siatki:
  - topologiczna – zawiera istotne miejsca, np. drzwi, skrzyżowania, ślepe zaułki. Rozdzielczość zależy od środowiska.
  - metryczna (uwagi: obserwacje mogą być różne dla różnych punktów pola siatki, model ruchu może się zmieniać w zależności od wielkości pola i prędkości robota)

Filtry cząsteczkowe

- połączenie łańcuchów Markowa z metodą Monte Carlo
- najpopularniejsza wersja: FastSLAM
- śledzone jest jednocześnie K cząstek, każda na swojej ścieżce
  - na podstawie odometrii generowane są nowe pozycje cząstek zgodnie z przyjętym rozkładem błędów

$$x_t^{[k]} \sim p(x_t | x_{t-1}^{[k]}, u_t)$$

  - przy nowym pomiarze wyliczane są *współczynniki istotności* na podstawie prawdopodobieństwa pomiaru cechy n w założonej pozycji

$$w_t^{[k]} = \mathcal{N}(z_t | x_t^{[k]}, \mu_{t,n}^{[k]}, \Sigma_{t,n}^{[k]}),$$

    następnie zestaw cząstek jest zastępowany nowym z rozkładem wynikającym z unormowanych w_i.

Filtry cząsteczkowe – algorytm

$\mathcal{X}_t$: zbiór m cząstek w chwili t

$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$

for $k = 1$ to M do

  $x_t^{[k]} = motion\_model(x_{t-1}^{[k]}, u_t)$

  $w_t^{[k]} = measurement\_model(z_t, x_t^{[k]}, m)$

  $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + (x_t^{[k]}, w_t^{[k]})$

endfor

for $k = 1$ to M do

  losuj $x_t^{[i]}$ z prawdopodobieństwem $\propto w_t^{[k]}$

  dodaj $\mathcal{X}_t = \mathcal{X}_t + x_t^{[i]}$

endfor

FastSLAM – algorytm

Wykorzystuje twierdzenie Rao-Blackwella(-Kołmogorowa). Cząstki mają postać

$$y_t^{[i]} = \left(x_{1:t}^{[i]} \ , \ \{\mu_j^{[i]}, \Sigma_j^{[i]}\} \right)$$

Dla każdej z M cząstek

- Wyciągnij posturę $x_{t-1}^{[i]}$ ze zbioru cząstek Y_{t-1}
- (predykcja) Wyznacz nową posturę $x_t^{[i]} \sim p(x_t | x_{t-1}^{[i]}, u_t)$
- (aktualizacja pomiarów) Dla każdej obserwacji z_t^k znajdź dopasowanie do cechy j i wprowadź z_t^k do odpowiedniego EKF, aktualizując $(\mu_{j,t}^{[i]}, \Sigma_{j,t}^{[i]})$
- (aktualizacja wag) Oblicz wagę ważności $w^{[i]}$ dla nowej cząstki

(resampling) Zastąp zbiór cząstek nowym, losowanym z prawdopodobieństwem proporcjonalnym do $w^{[i]}$

Metody MC

Rozszerzenia

- AugmentedMCL – dodawanie losowych próbek w liczbie zależnej od krótko- i długookresowej estymaty prawdopodobieństwa
- MixtureMCL – cześć cząstek generowana z modelu ruchu, a część z modelu obserwacji
- KLDSampling – dobór liczby cząstek na podstawie dywergencji Kullbacka-Leiblera

Środowiska dynamiczne

- rozszerzenie stanu – uwzględnienie w estymacie pozycji i prędkości ruchomych obiektów (por. HMM)
- outlier rejection – odrzucenie niewłaściwych pomiarów już w modelu czujnika

Podsumowanie metod

	EKF	MHT	grid (top.)	grid (met.)	MCL
obserwacje	cechy	cechy	cechy	surowe	surowe
zakłócenia	gaussowskie	gaussowskie	dowolne	dowolne	dowolne
wynik	$\mathcal{N}(\mu, \Sigma)$	$\{\mathcal{N}_i(\mu, \Sigma)\}$	histogram	histogram	cząstki
wydajność pamięciowa	$++$	$++$	$+$	-	$+$
wydajność czasowa	$++$	$+$	$+$	-	$+$
łatwość implementacji	$+$	-	$+$	-	$++$
rozdzielczość	$++$	$++$	-	$+$	$+$
odporność	-	$+$	$+$	$++$	$++$
globalna	nie	tak	tak	tak	tak