Estymacja Stanu

Uwagi

Proszę śledzić równocześnie slajdy bez notatek (aby nie widzieć od razu odpowiedzi na pytania). Gdy pojawią się zadania proszę się zatrzymać i postarać się je od razu zrobić (z reguły wyliczyć konkretne wartości), a następnie sprawdzić odpowiedzi (z reguły odpowiedzi i komentarze zamieściłem pod pytaniem w tym skrypcie, czasem rozwiązania jest po prostu na kolejnych slajdach lub na slajdach z notatkami).

W poniższym tekście nie jest wprost powiedziane do którego konkretnie slajdu dany fragment się odnosi ale powinno to być oczywiste z kontekstu.

Wstęp

W tym wykładzie opowiemy sobie o problemie estymacji stanu oraz trzech metodach które można wykorzystać do rozwiązania tego problemu.

W trakcie wykładu skupimy się na specyficznym problemie estymacji stanu a mianowicie na problemie lokalizacji. Polega on na tym, że chcemy znać dokładną pozycję naszego agenta w sytuacji gdy nie mamy sensorów które wprost mówiłyby nam gdzie agent się znajduje.

Dlaczego określenie lokalizacji agenta może być trudne? Z jakich cech środowiska może to wynikać?

Czy obserwowalność środowiska ma wpływ na problem lokalizacji? Tak. Przykładowo osobie niewidomej znacznie trudniej jest określić swoją pozycję — musi ona polegać na innych zmysłach które nie dają tak dokładnej informacji jak wzrok. Zazwyczaj musi ona polegać na historii swoich obserwacji, np.: wcześniej byłem w kuchni, więc teraz idę korytarzem w kierunku salonu itp. Tak więc brak pełnej obserwowalności (proszę pamiętać, że chodzi tu o obserwację wszystkich istotnych parametrów z punktu widzenia zadania które wykonujemy, a nie bezwzględnie wszystkich) powoduje potrzebę wnioskowania na temat aktualnego stanu środowiska – tutaj lokalizacji agenta.

A czy determinizm środowiska wpływa na trudność problemu lokalizacji? Oczywiście tak – podobnie jak poprzednio. Proszę sobie wyobrazić sytuację, że agent w pewnym momencie zna perfekcyjnie swoją pozycję (np. ktoś mu to powiedział). Agent nic nie robi (intencjonalnie) ale środowisko jest tak skonstruowane, że działają na niego pewne trudne do przewidzenia siły (np. ruch wody gdy agent pływa) — w takiej sytuacji (kierunek/prędkość ruchu wody jest niedeterministyczny lub stochastyczny) agent już po krótkiej chwili nie może być pewny gdzie się znajduje.

Przykładem problemu lokalizacji może być zadanie wyznaczenia dokładnej pozycji pojazdu autonomicznego w przypadku gdy ma on co prawda super dokładne mapy ale za to niedokładny odbiornik GPS. W dawnych czasach gdy systemy zwiększające dokładność GPSu (np. system poprawek ze stacji referencyjnej) nie były dostępne dokładność odbiornika GPS mogła być na poziomie kilku metrów. Taka dokładność z punktu widzenia pojazdu autonomicznego jest nie do zaakceptowania – aby samochód mógł się utrzymać na swoim pasie ruchu potrzebna jest lokalizacja z dokładnością do pojedynczych centymetrów.

Na wykładzie powiemy sobie w jaki sposób agent w niedeterministycznym środowisku może na podstawie niedokładnych sensorów lepiej oszacowywać swoją lokalizację niż opierać się tylko i wyłącznie na bezpośrednich odczytach z sensorów.

Cały wykład skupia się co prawda na problemie pozycjonowania agenta (ponieważ jest on łatwy do zaprezentowania i wyobrażenia sobie) ale proszę cały czas pamiętać, że wszystko odnosi się równie dobrze do innych parametrów środowiska – nie tylko pozycji agenta. Stąd tytuł wykładu jest znacznie ogólniejszy „estymacja stanu". Zaprezentowane metody mogą być użyte do oszacowania przykładowo rzeczywistego stanu naładowania akumulatorów, liczby i rodzaju przeciwników w grze komputerowej, ale również wysokości zarobków sąsiada z naprzeciwka ;)

Podsumowując wstęp jeszcze raz podkreślę, że pomimo tego, że wszystkie przykłady w tym wykładzie mówią o „lokalizacji" to równie dobrze mogłyby mówić o dowolnym innym zestawie parametrów środowiska (czyli właśnie tytułowego „stanu"). Podobnie, przez „model ruchu" w przykładach chodzi o rozkład prawdopodobieństwa mówiący

gdzie agent może się znaleźć po wykonaniu akcji polegającej na przemieszczeniu się w jakimś kierunku. Natomiast w ogólności chodzi nie tyle o fizyczny ruch agenta ale o zmianę stanu środowiska (dowolnych jego parametrów, np. czy pojawia się brud w przykładzie z odkurzaczem, co robią piesi w problemie autonomicznej taksówki itd.)

Filtr histogramowy

Na początku skonkretyzujmy sobie przykładowe zadanie na którym przedstawimy działanie filtru histogramowego. Wyobraźmy sobie, że mamy korytarz w którym znajduje się robot (nasz agent). Początkowo agent nie wie w którym miejscu korytarza się znajduje — nie dostał jeszcze żadnej informacji z żadnego sensora, a z dostępnej wiedzy na temat konstrukcji środowiska nie da się przewidzieć gdzie początkowo robot się znajdzie. W takiej sytuacji jedynym sensownym wnioskiem może być to, że agent musi z równym prawdopodobieństwem przewidywać każdą możliwą lokalizację. Inaczej: każda pozycja robota w korytarzu jest tak samo prawdopodobna.

Żeby robot/agent mógł się odnaleźć potrzebne są mu jakieś „punkty orientacyjne" – coś co mógłby wykryć za pomocą swoich sensorów i co pomogłoby w jego zlokalizowaniu. Wyobraźmy sobie, że robot posiada detektor drzwi (zielone prostokąty na rysunku), tzn. jego sensor mówi że robot stoi przy drzwiach albo gdzieś indziej (przy ścianie). Sensor w naszym przykładzie nie potrafi odróżniać drzwi między sobą – jego odpowiedź jest binarna: „drzwi" lub „ściana". Jest to jedyna dostępna informacja na podstawie której agent musi się zlokalizować.

Gdy robot zaobserwuje „drzwi" to może zakładać, że znajduje się przy jednych z trzech drzwi na korytarzu (zakładamy, że agent zna rozkład drzwi w korytarzu czyli że ma dokładną mapę środowiska). Co więcej — skoro na samym początku musiał założyć, że każda lokalizacja na korytarzu jest tak samo prawdopodobna to teraz logicznie musi zakładać, że równie dobrze może być przy dowolnych drzwiach na korytarzu (oczywiście sytuacja się trywializuje kiedy na korytarzu mamy tylko jedne drzwi które wykrywa sensor). Dodatkowo, gdy sensor nie jest idealny (tzn. czasami daje błędny odczyt/myli się – bardzo często tak jest w praktyce) obecne przekonanie agenta o swojej lokalizacji nie wyklucza również pozycji przy ścianie (zakłada, że jest to mało prawdopodobne, jeśli sensor myli się rzadko). Tak więc przekonanie agenta o swojej pozycji zmienia się w przekonanie a posteriori (po wykonaniu pomiaru) — dla niektórych miejsc prawdopodobieństwo że tam znajduje się agent się zwiększa, a dla innych miejsc szanse się zmniejszają.

Teraz załóżmy, że agent postanowił przesunąć się w prawo (wykonał akcję „idź w prawo"). To powoduje, że po wykonaniu tego ruchu przekonanie agenta o swej aktualnej pozycji zmienia się w stosunku do poprzedniego przekonania. Teraz najbardziej prawdopodobne jest, że znajduje się na prawo od jakiś drzwi. Co bardzo istotne — taki ruch (ogólnie: zmiana stanu środowiska) jest z reguły również niepewny. Przykładowo: robot włączył silniki na 6 sekund ale raz przejedzie trochę dalej niż zakładane 2m a raz trochę bliżej (w zależności np. od oporów podłoża – inaczej pojedzie po parkiecie a inaczej po dywanie itp.) Tak więc dość pewne wcześniejsze przekonanie o tym, że robot znajduje się dokładnie przy jednych z 3 drzwi teraz zamienia się na bardziej rozmyte przekonanie, że znajduje się „około" 2m na prawo od któryś z drzwi (na rysunku „górki" stają się niższe i szersze). Na rysunku jest to rysunek „prior" (przed pomiarem).

Prześledźmy co się stanie gdy teraz robot dostanie kolejny (drugi) odczyt z sensora: i ponownie jest to „drzwi". Tym razem widzimy, że przekonanie agenta o swojej lokalizacji bardzo mocno się precyzuje (drugi rysunek „posterior"), ponieważ w naszym przykładzie tylko jedne drzwi mogły być teraz zaobserwowane jeśli wcześniej (2m na lewo) robot był również przy drzwiach.

Przedstawiliśmy sobie przed chwilą ideę filtru histogramowego. Podsumowując ją:

1. Agent ma jakiś początkowy stan przekonania o swojej lokalizacji/stanu środowiska (w przykładzie był to rozkład równomierny, ale w ogólności może być zupełnie dowolny — zależy to od naszej wiedzy na temat środowiska).

2. Agent dokonuje pomiaru: jego przekonanie o rzeczywistym stanie środowiska staje się pewniejsze (odczyt danych z sensorów powoduje, że czegoś się dowiaduje).

3. Agent wykonuje jakąś akcję: ponieważ wynik (konsekwencja) akcji może być niepewny więc stan przekonania agenta o tym jak teraz wygląda stan staje się mniej pewny (rozmywa się).

4. Agent ponownie dokonuje pomiaru czyli wracamy do punktu 2.

Teraz sprawdźmy jak dokładnie wyglądają dwa główne punkty czyli jak wyliczyć stan przekonania po wykonaniu pomiaru oraz po wykonaniu akcji.

Na rysunku widzimy dyskretny świat: korytarz ma 5 pól (pozycji w których może znaleźć się robot). Pierwsze, czwarte i piąte pole jest zielone (np. z drzwiami) a drugie i trzecie – czerwone (np. ze ścianą). Początkowo

agent nie wie gdzie się znajduje, więc początkowe przekonanie jest takie, że agent znajduje się na każdym polu z prawdopodobieństwem 20%.

Sensor robota rozpoznaje kolor i wykrył, że agent jest na polu czerwonym. Uwzględnimy fakt, że sensor nie działa idealnie: wiemy (np. z dokumentacji czujnika), że zwróci „czerwony" z prawdopodobieństwem 60% gdy faktycznie jest na polu czerwonym, ale jednocześnie z prawdopodobieństwem 20% może zwrócić „czerwony" gdy w rzeczywistości znajduje się na polu zielonym.

Aby wyliczyć nowy stan przekonania musimy pomnożyć aktualne (a priori) wartości przez wartości 0.6 lub 0.2 w zależności czy odnosimy się odpowiednio do pola czerwonego czy do zielonego. Przypomnijmy — wartości 0.6 i 0.2 biorą się z dokumentacji działania sensora oraz samego wyniku pomiaru. Na rysunku pokazane są odpowiednie wyniki czyli wartości 0.04 lub 0.12. Jednak takie wartości nie sumują się do jedynki więc jeśli chcemy aby wartości przekonania można było interpretować w kategorii prawdopodobieństwa trzeba znormalizować te „surowe" wartości (czyli podzielić przez ich sumę). Jako ćwiczenie proszę wykonać obliczenia (zadanie 1).

Odpowiedź:

.

.

.

.

.

prawdopodobieństwo, że agent znajduje się na polu czerwonym to jedna trzecia, a że na polu zielonym to jedna dziewiąta.

Teraz sprawdźmy co się dzieje gdy agent wykonuje akcję. W naszym przykładzie zakładamy cykliczność świata, tzn. skrajnie prawe pole na rysunku sąsiaduje również ze skrajnie lewym polem. Tak więc stojąc na polu piątym i wykonując ruch w prawo agent przechodzi na pole pierwsze. Proszę się teraz zastanowić jak w takim razie będzie wyglądało przekonanie agenta o swojej pozycji gdy wykona ruch o jedno pole w prawo (zadanie 2).

Odpowiedź:

.

.

.

.

.

jedna dziewiąta, jedna dziewiąta, jedna trzecia, jedna trzecia, jedna dziewiąta — po prostu przesuwamy (cyklicznie) wszystkie wartości o jedno pole w prawo.

Rozważmy teraz bardziej rzeczywisty przypadek, gdy konsekwencje akcji są stochastyczne: przyjmijmy, że gdy agent chce wykonać ruch na jakieś pole to trafi na nie z prawdopodobieństwem 80% natomiast z prawdopodobieństwem po 10%, że trafi na pole z lewej albo z prawej strony docelowego pola. Przykładowo gdy, agent wykonuje akcję przejścia dwa pola w prawo to faktycznie może znaleźć się na pierwszym polu po prawej (z prawd. 10%), na drugim polu z prawej (zamierzony ruch: 80%) albo na trzecim polu po prawej (z prawd. 10%). Podobnie gdy agent wykonuje ruch o 1 pole w prawo, to z prawd. 10% pozostanie na starym miejscu itd.

Zadanie 3: jak przy powyższym modelu ruchu („model ruchu" to zadany rozkład prawdopodobieństwa, że agent znajdzie się na poszczególnych polach) i początkowym przekonaniu, że na 100% agent znajduje się na drugim polu będzie wyglądał rozkład po wykonaniu jednego ruchu o 2 pola w prawo?

Odpowiedź:

.

.

.

.

.

0, 0, 0.1, 0.8, 0.1 — mam nadzieję, że odpowiedź jest oczywista :)

Teraz trochę trudniejsze pytanie: jak rozkład będzie wyglądał gdy początkowe przekonanie jest takie, że na 50% jesteśmy w polu 2 a na 50% w polu 4. Proszę to sobie obliczyć! (zadanie 4)

Wskazówka: dla każdego pola będzie trzeba zsumować kilka wartości.

Zadanie 5: proszę się zastanowić jak w nieskończoności będzie wyglądał rozkład po ciągłym wykonywaniu ruchu o jedno pole w prawo (z modelem ruchu jak poprzednio czyli: 0.1, 0.8, 0.1)?

Odpowiedź:

.

.

.
.
.
rozkład z każdym ruchem będzie zbliżał się do rozkładu jednorodnego.

Kolejne slajdy przypominają, że proces lokalizacji rozpoczyna się od jakiegoś początkowego przekonania o stanie (wynikającego z naszej wiedzy o problemie) a następnie cyklicznie powtarza się odczyt z sensorów i ruch (wykonanie akcji przez agenta). Podczas odczytu danych z sensorów agent dowiaduje się czegoś o środowisku (entropia przekonania agenta maleje), natomiast podczas wykonania akcji agent traci informacje (entropia przekonania agenta rośnie).

Podsumowując: stan wiedzy agenta reprezentujemy dyskretnym rozkładem prawdopodobieństwa (jakie jest prawd. danego stanu). Podczas obserwacji wykonujemy mnożenie wartości rozkładu przez wartości wynikające z pomiaru oraz normalizujemy wyniki aby ponownie uzyskać rozkład prawdopodobieństwa. Natomiast podczas ruchu wykonuje się konwolucję obecnego stanu wiedzy z modelem ruchu.

Na koniec części dotyczącej filtru histogramowego spójrzmy na matematyczny/probabilistyczny opis problemu.

Najpierw obserwacja. Mamy tutaj dwie zmienne losowe: X mówiącą o pozycji robota (nasz stan przekonania), oraz Z — będącej wynikiem pomiaru (proszę pamiętać, że w ogólności pomiar może być niepewny więc modelujemy go też jako zmienną losową).

To co nas interesuje to rozkład prawdopodobieństwa X (czyli lokalizacji agenta) pod warunkiem zaobserwowania Z, co (korzystając z wzoru Bayesa – patrz poprzedni wykład) można zapisać jako zależność trzech rozkładów:

- rozkładu pozycji a priori (czyli przekonanie agenta przed dokonaniem odczytu z sensorów),
- rozkładu prawd. pomiaru (mówi jak działa sensor, np. z jakim prawd. sensor wykryje drzwi pomimo, że agent nie stoi przed żadnymi itd.)
- rozkładu prawd. wyników pomiarów — normalizacja z wcześniejszych slajdów.

Teraz ruch. Aby opisać rozkład prawd. po ruchu potrzebujemy rozkładu prawd. pozycji agenta przed ruchem, oraz rozkład prawd. związanego z naszą wiedzą na temat ruchu (model ruchu), tzn. prawd. znalezienia się na polu i pod warunkiem, że było się na polu j.

Podsumowując filtr histogramowy warto zwrócić uwagę na kilka jego cech:

- jest dyskretny, tzn. opis stanu jeśli naturalnie nie jest dyskretny to trzeba dokonać jego kwantyzacji,
- jest multimodalny — umożliwia reprezentowanie dowolnego rozkładu prawd. co jest istotną zaletą.

Filtra Kalmana

W praktyce głównym ograniczeniem w stosowaniu filtru histogramowego jest konieczność zapamiętania prawd. dla każdego możliwego stanu, co oznacza ogromne zapotrzebowanie na pamięć i czas potrzebny na aktualizację wszystkich wartości. Przykładowo załóżmy, że chcemy modelować pozycję pociągu na trasie Poznań–Warszawa. Jest to ok. 300km, czyli gdybyśmy chcieli modelować jego pozycję z dokładnością do 1m to do opisu naszego przekonania (rozkład prawd.) potrzebowalibyśmy 300 tysięcy wartości! Sytuacja staje się gorsza gdy do opisu pozycji potrzebnych jest więcej wartości, np. gdy agentem jest autonomiczny dron jego pozycję 3D możemy opisać trzema wartościami co oznacza, że potrzebujemy pamiętać i aktualizować wartości dla $(liczba\,różnych\,pozycji\,x) \times (liczba\,różnych\,pozycji\,y) \times (liczba\,różnych\,pozycji\,z)$. Tak więc wraz ze wzrostem wymiarowości sytuacja staje się coraz gorsza. W praktycznym przypadku drona, poza 3 współrzędnymi agent potrzebuje jeszcze wiedzieć jak jest przechylony w 3 osiach, co daje już 6 wymiarów. Przydałaby się też informacja jak te wszystkie wartości zmieniają się w czasie (prędkości) co daje już 12 wymiarów, gdy dla autopilota potrzebne będą również przyspieszenia to mamy już 18 wymiarów itd. Takiej przestrzeni stanów nie da się zapamiętać bezpośrednio jako dyskretny rozkład prawd. Dlatego dla takich problemów trzeba zastosować inne rozwiązania, np. filtr Kalmana.

Filtr Kalmana modeluje przekonanie agenta o swojej lokalizacji za pomocą rozkładu normalnego. Oznacza to, że zmienne systemu mogą być ciągłe, natomiast rozkład nie może być już zupełnie dowolny tak jak w filtrze histogramowym (na rysunku: „monte carlo localization") tylko musi być unimodalny (tzn. tylko jedna wyróżniona hipoteza ma największe prawdopodobieństwo).

Na kolejnym slajdzie dla przypomnienia widzimy przykładowe rozkłady normalne wraz ze wzorem jego funkcji gęstości. Warto tutaj zwrócić uwagę, że im mniejsza wariancja tym bardziej „stromy" rozkład co oznacza, że jesteśmy bardziej pewni jakiejś konkretnej wartości (pozycji). Im wariancja większa tym, bardziej prawd. że faktyczna pozycja leży gdzieś dalej od średniej (μ).

Schemat działania filtru Kalmana jest analogiczny jak filtru histogramowego. Naprzemiennie następuje pomiar (czyli odczyt z sensorów) oraz ruch (zmiana środowiska). Teraz pytanie kontrolne (zadanie 6) czy pamiętają Państwo jak było w filtrze histogramowym (tutaj jest identycznie): czy do zamodelowania/zaktualizowania rozkładu po wykonaniu pomiaru stosujemy iloczyn (wzór bayesa) czy konwolucję (wzór na prawdopodobieństwo całkowite)?

Odpowiedź:

.
.
.
.
.

oczywiście to pierwsze — wygląda to tak samo jak w filtrze histogramowym tylko rozkłady mamy inne.

Dla ruchu oczywiście trzeba wyznaczyć prawdopodobieństwo całkowite.

Kilka kolejnych slajdów dotyczą modyfikacji rozkładu podczas pomiaru. Na rysunku widzimy dwa przykładowe rozkłady: czarny rozkład a priori o średniej μ oraz niebieski rozkład lokalizacji wg. sensora (o średniej ν).

Proszę się zastanowić gdzie w takim razie będzie średnia nowego rozkładu (nowy stan przekonania agenta o swojej pozycji) po wykonaniu tego pomiaru (zadanie 7)?

Odpowiedź:

.
.
.
.
.

będzie ona gdzieś pomiędzy średnimi jednego i drugiego rozkładu czyli pomiędzy μ a ν.

Teraz proszę się zastanowić gdzie będzie amplituda funkcji gausowskiej po pomiarze (zadanie 8)?

Odpowiedź:

.
.
.
.
.

będzie ona wyżej od obydwu amplitud (zarówno od rozkładu czarnego jak i niebieskiego). Odpowiedź na poprzednie pytanie z reguły nie sprawia nam problemu (jest intuicyjna), natomiast odpowiedź na to pytanie jest dla większości osób zaskakująca. Musimy jednak pamiętać, że po wykonaniu pomiaru agent dowiaduje się czegoś nowego o świecie (posiada starą wiedzę + nową wiedzę) a więc będzie bardziej pewny swego nowego przekonania. Entropia stanu przekonania maleje czyli w rozkładzie normalnym maleje wariancja (w konsekwencji amplituda jest wyższa).

Wzory jak wyliczyć nową średnią oraz wariancję podane są na kolejnym slajdzie. Widać z nich, że nowa średnia jest bliższa temu rozkładowi który miał mniejszą wariancję. Natomiast nowa wariancja jest równa $\frac{r^2\sigma^2}{r^2+\sigma^2}$ z czego jasno widać, że jest mniejsza od wariancji zarówno rozkładu a priori jak i rozkładu pomiaru.

Jako proste ćwiczenie proszę szybko policzyć w pamięci zadanie 9.

Na kolejnym slajdzie widzimy krótkie podsumowanie pomiaru pokazujące odpowiednie prawd. warunkowe dwóch zmiennych losowych: lokalizacji a priori oraz pomiaru.

Zadanie 10: jak będzie wyglądał rozkład a posteriori?

Odpowiedź:

.
.
.
.
.

prawidłowa odpowiedź to zielony rozkład – mam nadzieję, że odpowiedź była już teraz oczywista. Proszę zwrócić uwagę, na pewną charakterystyczną cechę filtru Kalmana. Wyobraźmy sobie, że mamy system nawigacji GPS i zbliżamy się na autostradzie do zjazdu. W takiej sytuacji nasza nawigacja może domniemywać, że ciągle jedziemy autostradą, GPS natomiast może pokazać, że już zjeżdżamy z autostrady. Co ciekawe, filtr Kalmana pokaże w takiej sytuacji, że z całą pewnością jesteśmy w barierkach oddzielających obie drogi! Dlatego filtr Kalmana nadaje się do sytuacji gdy rozkład a priori nie różni się zbytnio od wskazań sensorów, innymi słowy filtr Kalmana nadaje się do uwzględnienia pewnych niedokładności (sensorów oraz modelu ruchu) natomiast nie nadaje się do

integracji alternatywnych hipotez (bo rozkład normalny jest unimodalny) — czegoś z czym filtr histogramowy nie miał żadnych problemów (bo może reprezentować dowolny rozkład prawd.)

Teraz przedstawmy sobie co się dzieje przy wykonaniu ruchu. W tym wypadku jest to chyba dość intuicyjne: wynikowa średnia jest sumą średniej rozkładu a priori (μ) oraz średniej z modelu ruchu (ν). Podobnie nowa wariancja jest równa sumie wariancji obu tych rozkładów — zwróćmy uwagę, że wynikowa wariancja się zwiększa czyli agent po wykonaniu ruchu jest mniej pewny faktycznego stanu (po ruchu tracimy informację — entropia rośnie).

Do tej pory omawialiśmy sobie prosty filtr Kalmana dla systemu z jedną zmienną (lokalizacja w jednym wymiarze). Oczywiście filtr Kalmana można zastosować dla problemów wielowymiarowych (stan przekonania jest wtedy wielowymiarowym rozkładem normalnym). Co ciekawe filtr umożliwia przewidywanie jednej ze zmiennych stanu środowiska na podstawie tylko sensora mierzącego wartość na innym wymiarze (dzięki znajomości modelu ruchu).

Przykładowo załóżmy, że stan można opisać za pomocą pozycji 1D (odległość pociągu od stacji X) oraz jego prędkości.

Kolejne 2 slajdy to przypomnienie wielowymiarowej funkcji gaussowskiej.

Ok, wracamy do przykładu – zakładamy kolejne kroki czasowe $t = 1, 2, 3, \ldots$. Na osi odciętych mamy pozycję agenta, a na osi rzędnych jego prędkość. Początkowo zakładamy, że wiemy iż agent jest na pozycji 1, natomiast nie wiemy jaką ma prędkość. Ponieważ w filtrze Kalmana używamy wielowymiarowego rozkładu normalnego więc taką sytuację trzeba przedstawić w ten sposób że wariancja prędkości jest nieskończona (w praktyce: baaardzo duża). Jakby ten wykres wyglądał gdyby znana prędkość wynosiła np. 0 albo 1?

Odpowiedź:

.

.

.

.

.

niebieski rozkład nie byłby taki „pionowy” tylko zbliżony do okręgów wycentrowanych na prędkości równej zero lub jeden.

Teraz przejdźmy do kolejnej chwili czasowej $t = 2$. Ponieważ jeszcze nie wykonaliśmy pomiaru musimy dokonać predykcji jak mógł zmienić się stan środowiska pomiędzy chwilą $t = 1$ a $t = 2$. Na rysunku czerwony rozkład reprezentuje przekonanie agenta gdzie może się znajdować i jaką może mieć prędkość przy założeniu, że nowa pozycja to stara przesunięta zależnie od prędkości ($x' = x + \dot{x}$). Czerwony obszar pokazuje np. że gdyby prędkość była równa zero (oś rzędnych) to pozycja (oś odciętych) byłaby w okolicach 1. Gdyby prędkość była równa mniej więcej 1 to pozycja byłaby w okolicach 2, itd.

Teraz następuje pomiar pozycji z którego wynika, że agent jest mniej więcej na pozycji 2 (zielony rozkład reprezentujący sam pomiar jest taki wysoki, ponieważ reprezentuje on pomiar który zupełnie nie mierzy prędkości — stąd duża wariancja na osi prędkości). Czarny obszar reprezentuje wynikowy rozkład leżący na przecięciu wiedzy a priori (czerwony rozkład) oraz pomiaru (zielony rozkład). Widać z niego, że po wykonaniu tego pomiaru agent jest nie tylko przekonany o swojej obecnej pozycji ale również o prędkości której nigdy nie mierzył.

Jak to możliwe? Otóż projektując filtr Kalmana trzeba podać m.in. funkcję przejścia mówiącą jak na podstawie stanu systemu w kroku t będzie wyglądał stan w kroku $t + 1$. Funkcja ta jest wyidealizowana (w naszym przykładzie zakłada ona np. ruch jednostajny), wszelkie niedokładności filtr uwzględnia w innych parametrach systemu mówiących m.in. o niepewności czy dokładności funkcji przejścia. Drugim koniecznym elementem jest podanie funkcji pomiaru mówiącej co nasze sensory powinny zwrócić dla danego faktycznego stanu systemu. I tutaj znów wszelkie niepewności uwzględnia się w odpowiednich parametrach filtru.

Kolejny slajd pokazuje równania reprezentujące obliczenia które się wykonuje na kolejnych etapach w trakcie działania filtru (nie trzeba ich znać na kolokwium natomiast warto je prześledzić). Warto wiedzieć, że istnieje również rozszerzony filtr Kalmana który obsługuje nieliniowe systemy (na slajdach zarówno funkcja przejścia jak i funkcja pomiaru są liniowe).

Filtr cząsteczkowy

Filtr Kalmana rozwiązywał 2 główne problemy filtru histogramowego: obsługiwał ciągłą przestrzeń stanów oraz był wydajny obliczeniowo. Natomiast jego wadą jest unimodalność. Ograniczenie się tylko do wielowymiarowych rozkładów normalnych jest bez wątpienia bardzo istotnym problemem ograniczającym jego praktyczne zastosowania.

Dlatego omówimy sobie też filtr cząsteczkowy który umożliwia reprezentowanie ciągłych przestrzeni stanów i jednocześnie multimodalnych rozkładów prawdopodobieństw (przekonań agenta o stanie systemu). Co istotne filtr cząsteczkowy jest prosty w implementacji i bardzo elastyczny jeśli chodzi o ustawienia przetargu między jego

dokładnością (w sensie dokładności reprezentowania rozkładu prawd.) a wymagania dot. prędkości działania i potrzebnej pamięci.

Rozważmy przykładowe środowisko: nasz agent jest robotem który może poruszać się do przodu oraz skręcać w lewo/prawo. Robot ma sensory mówiące w jakiej odległości znajduje się od pewnych charakterystycznych punktów terenu.

Ideą filtru cząsteczkowego jest reprezentowanie rozkładu prawd. za pomocą „cząsteczek" które odpowiadają różnym hipotezom (w filtrze histogramowym mieliśmy niejako po jednej „cząsteczce" na każdą możliwą hipotezę, tzn. dla każdego możliwego stanu pamiętaliśmy jego prawdopodobieństwo). Im więcej będziemy mieli cząsteczek tym dokładniej będziemy mogli reprezentować rozkład prawdopodobieństwa (zbliżamy się z dokładnością do filtru histogramowego). Im mniej tym filtr będzie bardziej podobny do filtru Kalmana.

Pierwszym krokiem algorytmu jest zainicjowanie N cząstek losowo – tzn. w naszym przykładzie każdej cząstce przypisujemy losową pozycję i obrót robota (ponieważ znowu nic nie wiemy o początkowej lokalizacji robota w środowisku).

W drugim kroku symulujemy ruch robota: każdą cząstkę (hipotezę) aktualizujemy wg. zamierzonego ruchu (akcji wykonanej przez agenta), czyli przesuwamy oraz obracamy „cząstkę" (reprezentującą agenta) zgodnie z zamierzoną akcją. Aby algorytm nie zbiegł się do punktu (zdegenerowany rozkład) dodajemy „trochę" szumu (w naszym przykładzie do pozycji oraz kąta).

Aby filtr działał musimy każdej cząstce przypisać pewną wagę mówiącą jak dobra jest dana hipoteza. Robi się to na podstawie danych z sensorów (musimy się czegoś dowiedzieć o świecie, inaczej nie mamy podstaw aby twierdzić, że jedna hipoteza jest lepsza od innej) — jeśli pomiar zgadza się z daną hipotezą to cząstka dostaje dużą wagę, gdy pomiar niezbyt pasuje do hipotezy to cząstka dostaje odpowiednio mniejszą wagę.

Jak oblicza się wagę? Dla każdego punktu odniesienia L_j (charakterystyczny punkt terenu) sensor mierzy odległość i zwraca wynik w postaci rozkładu normalnego: średnia mówi jaką odległość zmierzył od robota do punktu odniesienia, a wariancja mówi o pewności dokonanego pomiaru. Z drugiej strony wiemy jaka jest dokładnie odległość danej cząstki p_i (hipotezy) od punktu L_j ($dist(p_i, L_j)$). Wagą z punktu widzenia pojedynczego punktu L_j będzie wartość z tego rozkładu normalnego dla odległości $dist(p_i, L_j)$ — czyli im odległość $dist(p_i, L_j)$ jest bliższa zmierzonej (na rysunku jest μ_i ale powinno być μ_j bo chodzi o odległość zmierzoną do j-tego punktu odniesienia) tym waga będzie większa. Dla wielu punktów odniesienia (L_j) wymnażamy wszystkie pojedyncze wartości. Krok 3 algorytmu formalizuje powyższy opis.

W kroku 4 algorytmu próbkuje się cząstki wg. ich wag — losuje się (z powtórzeniami) nową populację cząstek która będzie bardziej skupiona w rejonach odpowiadających hipotezom zgodnym z faktycznymi pomiarami. Prawdopodobieństwo wylosowania danej cząstki jest proporcjonalne do wagi tej cząstki. Im cząstka miała mniejszą wagę (czyli hipoteza nie zgadzała się z fizycznym pomiarem) tym mniejszą szansę będzie miała pozostać w naszej populacji potencjalnych hipotez.

Jako ćwiczenie proszę policzyć prawdopodobieństwa NIEwylosowania cząstki p_3 (zadanie 11) i p_1 (zadanie 12) przy założeniu, że w filtrze mamy 5 cząstek z zadanymi wagami.

Losowanie cząsteczek wg. ich wag przebiega jak w selekcji ruletkowej. Tzn. możemy sobie wyobrazić, że dzielimy koło na fragmenty proporcjonalne do wag każdej cząstki, kręcimy kołem i który fragment nam wypadnie to do nowej populacji bierzemy odpowiadającą mu cząsteczkę. Proszę się zastanowić jak można to zaimplementować? Jaką złożoność ma taki algorytm? Prawdopodobnie będzie to algorytm w którym wybór pojedynczej cząstki ma złożoność $O(N)$ (wybór liniowy) albo $O(logN)$ (przeszukiwanie binarne). Istnieją jednak algorytmy o mniejszej złożoności ($O(1)$) – warto o tym pamiętać i w razie potrzeby takie zastosować.

Pytanie: czy w naszym przykładzie z robotem orientacja cząstek ma w ogóle znaczenie? (zadanie 13)

Odpowiedź:

.

.

.

.

.

w drugim kroku algorytmu symuluje się ruch robota i tam orientacja ma znaczenie bo przesuwamy się w danym kierunku. Nie ma natomiast bezpośredniego udziału w ważeniu cząsteczki — sytuacja byłaby inna gdyby nasze sensory były czułe na kierunek robota (ale w przykładzie zwracają tylko odległość robot–punkt charakterystyczny).

Podsumowując filtr cząsteczkowy warto zwrócić uwagę, że pomimo prostej idei a więc łatwości implementacji filtr jest bardzo skuteczny w wielu przypadkach (nawet w przypadku gdy istotne jest modelowanie bardzo skomplikowanych rozkładów). Niestety wadą filtru jest problem pojawiający się w przypadku wielu wymiarów (przekleństwo wymiarowości). Im jest ich więcej tym więcej (wykładniczo względem liczby wymiarów!) potrzebujemy cząstek aby

równomiernie rozłożyć cząsteczki w przestrzeni (równomiernie próbkować możliwe hipotezy). Pewnym złagodzeniem tego problemu jest trywialna możliwość zwiększenia liczby cząstek — niestety dla wielu wymiarów natkniemy się na problem zasobów obliczeniowych i będziemy musieli pogodzić się z mniej dokładnym modelowaniem rzeczywistego rozkładu.