Agenty reaktywne i hybrydowe (*Reactive and hybrid agents*)

Na podstawie "An Introduction to MultiAgent Systems" oraz slajdów Michaela Wooldridge'a

Architektura reaktywna - subsumpcyjna

- Problem z podejściami symbolicznymi/logicznymi (transdukcja, złożoność obliczeniowa) $\rightarrow$ paradygmat reaktywny

Architektura reaktywna - subsumpcyjna

- Problem z podejściami symbolicznymi/logicznymi (transdukcja, złożoność obliczeniowa) $\rightarrow$ paradygmat reaktywny
- Najbardziej znana architektura reaktywna oparta na następujących założeniach
  - inteligentne zachowanie można wygenerować bez jawnej reprezentacji
  - inteligentne zachowanie można wygenerować bez jawnego wnioskowania
  - inteligentne zachowanie to (samoczynnie) pojawiająca się właściwość pewnych złożonych systemów

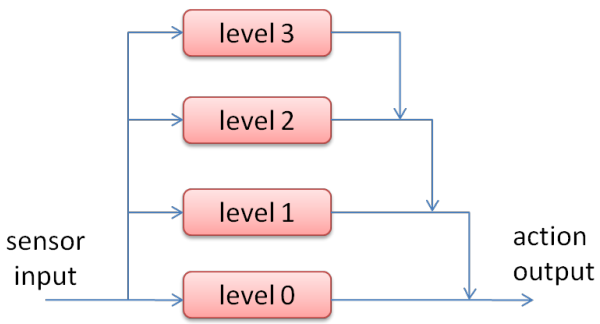
Charakterystyka architektury subsumpcyjnej

- Podejmowanie decyzji zrealizowane poprzez zbiór odpowiednich zachowań
  - każde zachowanie to funkcja, która mapuje percept ze środowiska na akcję do wykonania
  - zachowania reprezentowane jako reguły bez wykorzystania reprezentacji symbolicznej (percept $\rightarrow$ akcja)

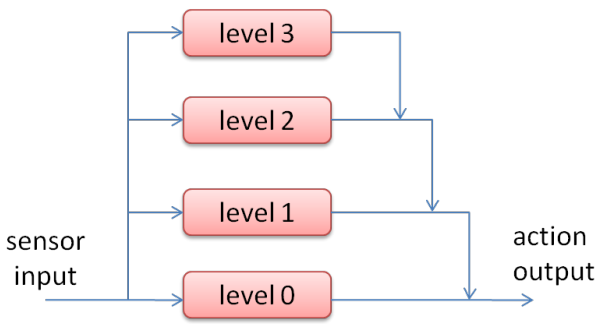
Charakterystyka architektury subsumpcyjnej

- Podejmowanie decyzji zrealizowane poprzez zbiór odpowiednich zachowań
  - każde zachowanie to funkcja, która mapuje percept ze środowiska na akcję do wykonania
  - zachowania reprezentowane jako reguły bez wykorzystania reprezentacji symbolicznej (percept $\rightarrow$ akcja)
- Wiele zachowań może być uruchamianych (wyzwalanych) jednocześnie
  - zachowania tworzą hierarchę subsumpcji podzielną na warstwy (im niższa warstwa, tym wyższy priorytet zachowania)
  - niże warstwy w hierarchii mogą zablokować wyższe warstwy

Wybór akcji w architekturze subsumpcyjnej

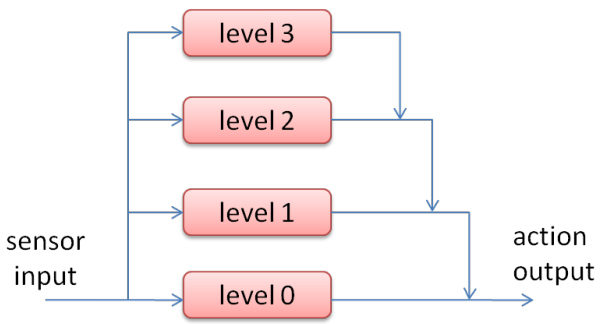


Wybór akcji w architekturze subsumpcyjnej



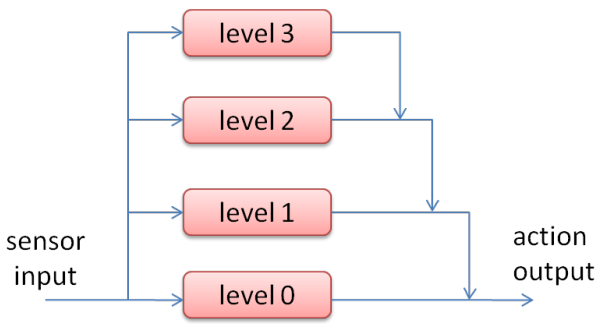
- „Surowy” odczyt z sensora nie jest ani przetwarzany, ani transformowany

Wybór akcji w architekturze subsumpcyjnej



- „Surowy” odczyt z sensora nie jest ani przetwarzany, ani transformowany
- Akcje zdefiniowane przez zbiór zachowań wraz z relacją blokowania

Wybór akcji w architekturze subsumpcyjnej



- „Surowy” odczyt z sensora nie jest ani przetwarzany, ani transformowany
- Akcje zdefiniowane przez zbiór zachowań wraz z relacją blokowania
- $b_1 \prec b_2$ oznacza, że 'b_1 blokuje b_2' – b_1 znajduje się niżej w hierarchii niż b_2 i przez to ma wyższy priorytet niż b_2

Mars Explorer

Naszym celem jest eksploracja odległej planety i zebranie próbek cennych skał. Położenie próbek nie jest znane – wiadomo tylko, że występują one w skupiskach. Mamy do dyspozycji zestaw autonomicznych robotów, które mogą poruszać się po planecie zbierając próbki i dostarczać je do statku-bazy. Nie mamy mapy planety – wiadomo tylko, że jest na niej dużo przeszkód, które uniemożliwiają komunikację radiową.

Wykorzystane mechanizmy

1. Pole gradientu
   - Statek-baza generuje sygnał radiowy, dzięki któremu agenty-roboty znają położenie statku
   - Sygnał nie niesie żadnej informacji
   - Aby dostać się do statku, agent musi poruszać się w stronę rosnącego gradientu

Wykorzystane mechanizmy

1. Pole gradientu
   - Statek-baza generuje sygnał radiowy, dzięki któremu agenty-roboty znają położenie statku
   - Sygnał nie niesie żadnej informacji
   - Aby dostać się do statku, agent musi poruszać się w stronę rosnącego gradientu
2. Pośrednia komunikacja
   Agenty wykorzystują „radioaktywne okruszki”, które mogą być upuszczane, podnoszone i wyczuwane przez przejeżdżające roboty

Zachowania indywidualne

- b_0: **if** detect an obstacle **then** change direction

Zachowania indywidualne

- b_0: **if** detect an obstacle **then** change direction
- b_1: **if** carrying a sample **and** at the base **then** drop sample

Zachowania indywidualne

- b_0: **if** detect an obstacle **then** change direction
- b_1: **if** carrying a sample **and** at the base **then** drop sample
- b_2: **if** carrying a sample **and not** at the base **then** travel up gradient

Zachowania indywidualne

- b_0: **if** detect an obstacle **then** change direction
- b_1: **if** carrying a sample **and** at the base **then** drop sample
- b_2: **if** carrying a sample **and not** at the base **then** travel up gradient
- b_3: **if** detect a sample **and not** at the base **then** pick up sample

Zachowania indywidualne

- b_0: **if** detect an obstacle **then** change direction
- b_1: **if** carrying a sample **and** at the base **then** drop sample
- b_2: **if** carrying a sample **and not** at the base **then** travel up gradient
- b_3: **if** detect a sample **and not** at the base **then** pick up sample
- b_4: **if** true **then** move randomly

Zachowania indywidualne

- b_0: **if** detect an obstacle **then** change direction
- b_1: **if** carrying a sample **and** at the base **then** drop sample
- b_2: **if** carrying a sample **and not** at the base **then** travel up gradient
- b_3: **if** detect a sample **and not** at the base **then** pick up sample
- b_4: **if** true **then** move randomly

Zachowania te tworzą następującą hierarchię:
$b_0 \prec b_1 \prec b_2 \prec b_3 \prec b_4$

Zachowania kooperatywne

- b_2^{new} (zamiast b_2): **if** carrying a sample **and not** at the base **then** drop 2 crumbs **and** travel up gradient

Zachowania kooperatywne

- b_2^{new} (zamiast b_2): **if** carrying a sample **and not** at the base **then** drop 2 crumbs **and** travel up gradient
- $b_{3.5}$ (pomiędzy b_3 and b_4): **if** sense crumbs **then** pick 1 crumb **and** travel down gradient

Zachowania kooperatywne

- b_2^{new} (zamiast b_2): **if** carrying a sample **and not** at the base **then** drop 2 crumbs **and** travel up gradient
- $b_{3.5}$ (pomiędzy b_3 and b_4): **if** sense crumbs **then** pick 1 crumb **and** travel down gradient

Zmodyfikowana hierarchia subsumpcji jest następująca:
$b_0 \prec b_1 \prec b_2^{new} \prec b_3 \prec b_{3.5} \prec b_4$

Ograniczenia agentów reaktywnych

1. Brak modeli środowiska – konieczność wykorzystywana rozbudowanej informacji lokalnej w celu wyboru akceptowalnych akcji

Ograniczenia agentów reaktywnych

1. Brak modeli środowiska – konieczność wykorzystywana rozbudowanej informacji lokalnej w celu wyboru akceptowalnych akcji
2. Brak perspektywy długookresowej w podejmowaniu decyzji

Ograniczenia agentów reaktywnych

1. Brak modeli środowiska – konieczność wykorzystywana rozbudowanej informacji lokalnej w celu wyboru akceptowalnych akcji
2. Brak perspektywy długookresowej w podejmowaniu decyzji
3. Brak jasnych związków między indywidualnymi zachowaniami a zachowaniem całego systemu $\rightarrow$ brak metodologii pozwalającej na budowę agentów realizujących konkretne (i zaawansowane) zadania

Ograniczenia agentów reaktywnych

1. Brak modeli środowiska – konieczność wykorzystywana rozbudowanej informacji lokalnej w celu wyboru akceptowalnych akcji
2. Brak perspektywy długookresowej w podejmowaniu decyzji
3. Brak jasnych związków między indywidualnymi zachowaniami a zachowaniem całego systemu $\rightarrow$ brak metodologii pozwalającej na budowę agentów realizujących konkretne (i zaawansowane) zadania
4. Trudności w przypadku budowy bardziej złożonych agentów (10 warstw i więcej) z uwagi na możliwe interakcje zachodzące pomiędzy różnymi zachowaniami

Pokaz

A teraz trochę zabawy... :)

Agenty hybrydowe

- Ekstremalne podejścia (tylko pro-aktywne lub tylko reaktywne) nie sprawdzają się w praktycznych problemach

Agenty hybrydowe

- Ekstremalne podejścia (tylko pro-aktywne lub tylko reaktywne) nie sprawdzają się w praktycznych problemach
- Połączenie obu podejść do budowy agentów składających się z 2 (lub więcej) podsystemów
  - podsystemu stosującego wnioskowanie praktyczne, zawierającego model środowiska oraz tworzącego plany z wykorzystaniem reprezentacji symbolicznych
  - podsystemu reaktywnego, który potrafi reagować na zdarzenia bez złożonego wnioskowania

Agenty hybrydowe

- Ekstremalne podejścia (tylko pro-aktywne lub tylko reaktywne) nie sprawdzają się w praktycznych problemach
- Połączenie obu podejść do budowy agentów składających się z 2 (lub więcej) podsystemów
  - podsystemu stosującego wnioskowanie praktyczne, zawierającego model środowiska oraz tworzącego plany z wykorzystaniem reprezentacji symbolicznych
  - podsystemu reaktywnego, który potrafi reagować na zdarzenia bez złożonego wnioskowania
- Podsystemy są zorganizowane w hierarchię komunikujących się ze sobą warstw ($\rightarrow$ architektury warstwowe)

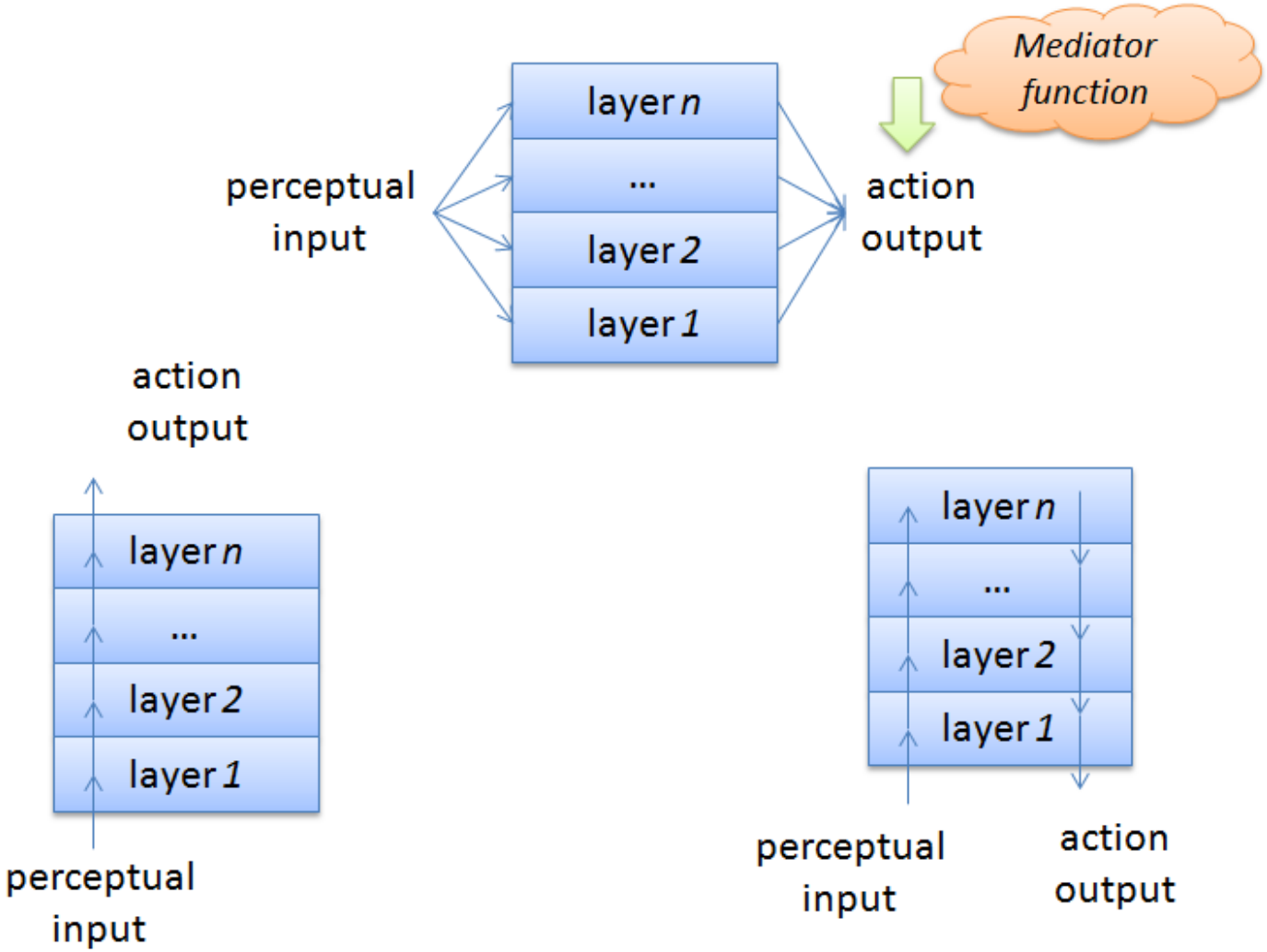
Typy architektur warstwowych

- Horyzontalne/równoległe (*horizontal layering*)
  Poszczególne warstwy są podłączone do sensorów i do aktuatorów – każda warstwa przypomina niezależnego agenta, który wybiera najbardziej odpowiednią akcję do wykonania

Typy architektur warstwowych

- Horyzontalne/równoległe (*horizontal layering*)
  Poszczególne warstwy są podłączone do sensorów i do aktuatorów – każda warstwa przypomina niezależnego agenta, który wybiera najbardziej odpowiednią akcję do wykonania
- Wertykalne/szeregowe (*vertical layering*)
  Sensory i aktuatory obsługiwane są przez jedną warstwę – przepływ sterowania miedzy poszczególnymi warstwami

Typy architektur warstwowych



TouringMachines

- TouringMachines składa się z trzech równoległych warstw

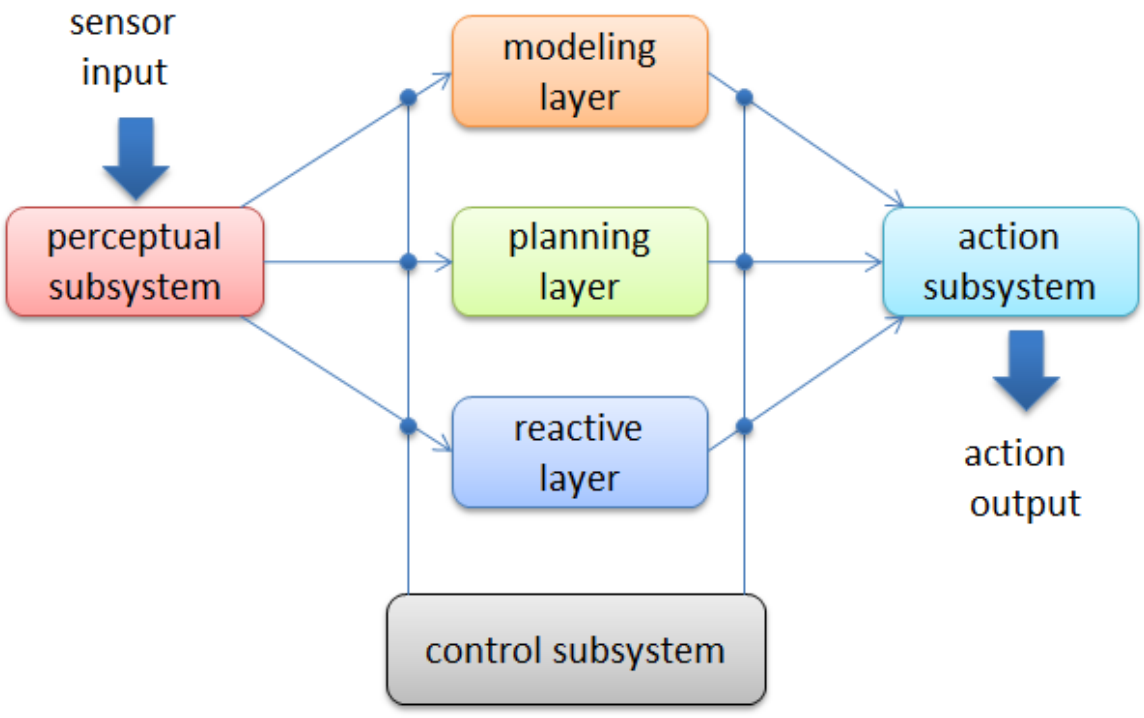
TouringMachines

- TouringMachines składa się z trzech równoległych warstw
- Każda warstwa dostarcza sugestii dotyczących akcji do wykonania przez agenta

TouringMachines

- TouringMachines składa się z trzech równoległych warstw
- Każda warstwa dostarcza sugestii dotyczących akcji do wykonania przez agenta
- Rozważany scenariusz – sterowanie autonomicznym pojazdem w środowisku, w którym porusza się wiele takich pojazdów (rozważania teoretyczne/symulacja)

Architektura TouringMachines



Warstwy w TouringMachines

- Warstwa reaktywna (*reactive*)
  - dostarcza natychmiastowych reakcji (akcji) na zmiany w środowisku
  - zaimplementowana jako zbiór reguł *sytuacja* $\rightarrow$ *akcja*, podobnie jak w architekturze subsumpcyjnej

Warstwy w TouringMachines

- Warstwa reaktywna (*reactive*)
  - dostarcza natychmiastowych reakcji (akcji) na zmiany w środowisku
  - zaimplementowana jako zbiór reguł *sytuacja* $\rightarrow$ *akcja*, podobnie jak w architekturze subsumpcyjnej

```
rule-1: kerb-avoidance
    if
        is-in-front(Kerb, Observer) and
        speed(Observer) > 0 and
        separation(Kerb, Observer) < KerbThreshHold
    then
        change-orientation(KerbAvoidanceAngle)
```

Warstwy w TouringMachines

- Warstwa planowania (*planning*)
  - odpowiada za proaktywne zachowanie agenta (formułowanie i wykonywanie planów)
  - tworzy finalne plany korzystając z biblioteki schematów/szkieletów planów
  - wybiera na bieżąco najbardziej odpowiedni plan

Warstwy w TouringMachines

- Warstwa planowania (*planning*)
  - odpowiada za proaktywne zachowanie agenta (formułowanie i wykonywanie planów)
  - tworzy finalne plany korzystając z biblioteki schematów/szkieletów planów
  - wybiera na bieżąco najbardziej odpowiedni plan
- Warstwa modelowania (*modeling*)
  - reprezentuje różne obiekty w środowisku (również samego agenta)
  - przewiduje konflikty między agentami i generuje cele/intencje w celu rozwiązania tych konfliktów $\rightarrow$ cele te są przekazywane do warstwy planowania

Podsytem sterowania w TouringMachines

- Decyduje, która warstwa powinna w danej sytuacji kontrolować agenta (wybrać akcję)

Podsytem sterowania w TouringMachines

- Decyduje, która warstwa powinna w danej sytuacji kontrolować agenta (wybrać akcję)
- Implementacja w formie zbioru reguł, które mogą blokować wejście z sensorów lub wyjście z poszczególnych warstw kontrolnych

Podsytem sterowania w TouringMachines

- Decyduje, która warstwa powinna w danej sytuacji kontrolować agenta (wybrać akcję)
- Implementacja w formie zbioru reguł, które mogą blokować wejście z sensorów lub wyjście z poszczególnych warstw kontrolnych

Podsytem sterowania w TouringMachines

- Decyduje, która warstwa powinna w danej sytuacji kontrolować agenta (wybrać akcję)
- Implementacja w formie zbioru reguł, które mogą blokować wejście z sensorów lub wyjście z poszczególnych warstw kontrolnych

```
censor-rule-1:
    if
        entity(obstacle-6) in perception-buffer
    then
        remove-sensory-record(layer-R, entity(obstacle-6))
```

InteRRaP

- InterRRaP wykorzystuje warstwy horyzontalne i dwukierunkowy przepływ sterowania miedzy nimi

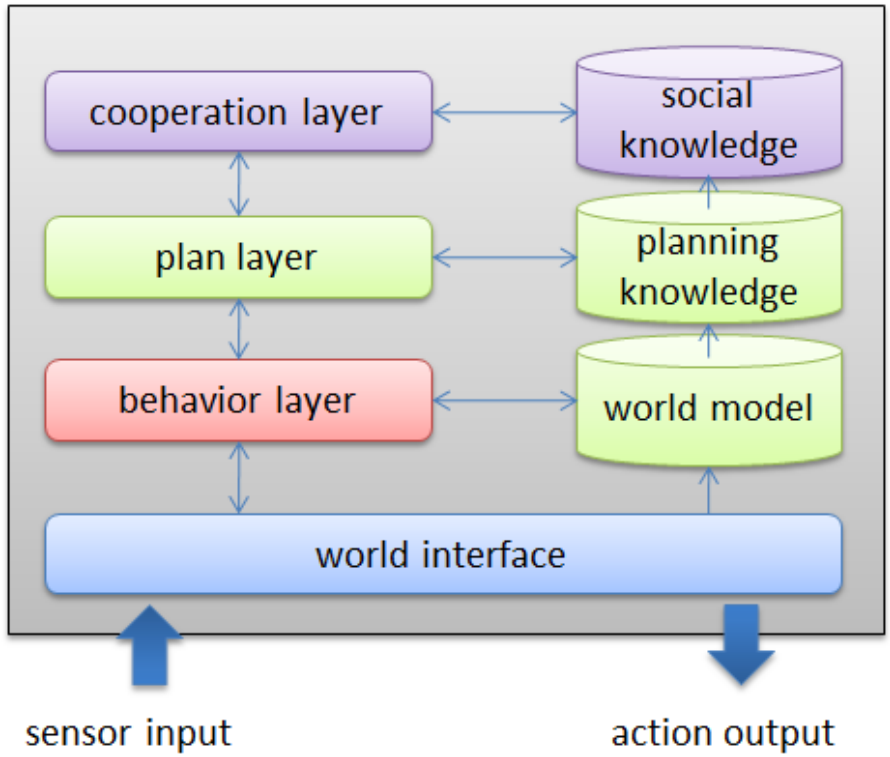
InteRRaP

- InterRRaP wykorzystuje warstwy horyzontalne i dwukierunkowy przepływ sterowania miedzy nimi
- Składa się z trzech warstw
  - *behavior-based* – odpowiedzialnej za zachowanie reaktywne
  - *local planning* – odpowiedzialnej za ustalanie planów i ich realizację
  - *cooperative planning* – odpowiedzialnej za interakcje ze środowiskiem

InteRRaP

- InterRRaP wykorzystuje warstwy horyzontalne i dwukierunkowy przepływ sterowania miedzy nimi
- Składa się z trzech warstw
  - *behavior-based* – odpowiedzialnej za zachowanie reaktywne
  - *local planning* – odpowiedzialnej za ustalanie planów i ich realizację
  - *cooperative planning* – odpowiedzialnej za interakcje ze środowiskiem
- Każda warstwa posiada swoja bazę wiedzy z reprezentacją środowiska odpowiednią dla danej warstwy – od „surowych” sygnałów z sensorów do złożonych modeli środowiska

Architektura InterRRaP



Interakcje między warstwami w InterRRaP

- Aktywacja *bottom-up* $\rightarrow$ niższa warstwa przekazuje sterowanie do warstwy wyższej jeśli nie ma wystarczających kompetencji, aby poradzić sobie z aktualną sytuacją

Interakcje między warstwami w InterRRaP

- Aktywacja *bottom-up* $\rightarrow$ niższa warstwa przekazuje sterowanie do warstwy wyższej jeśli nie ma wystarczających kompetencji, aby poradzić sobie z aktualną sytuacją
- Realizacja *top-down* $\rightarrow$ warstwa wyższego poziomu korzysta z usług dostarczonych przez warstwę poniżej

Interakcje między warstwami w InterRRaP

- Aktywacja *bottom-up* $\rightarrow$ niższa warstwa przekazuje sterowanie do warstwy wyższej jeśli nie ma wystarczających kompetencji, aby poradzić sobie z aktualną sytuacją
- Realizacja *top-down* $\rightarrow$ warstwa wyższego poziomu korzysta z usług dostarczonych przez warstwę poniżej
- Przepływ sterowania rozpoczyna się od odebrania sygnału z receptora przez najniższą warstwę, może dotrzeć do wyższych warstw i następnie wraca na dół

DARPA Challenge – Stanley

- Pierwsze miejsce w DARPA Grand Challenge w 2005 r.

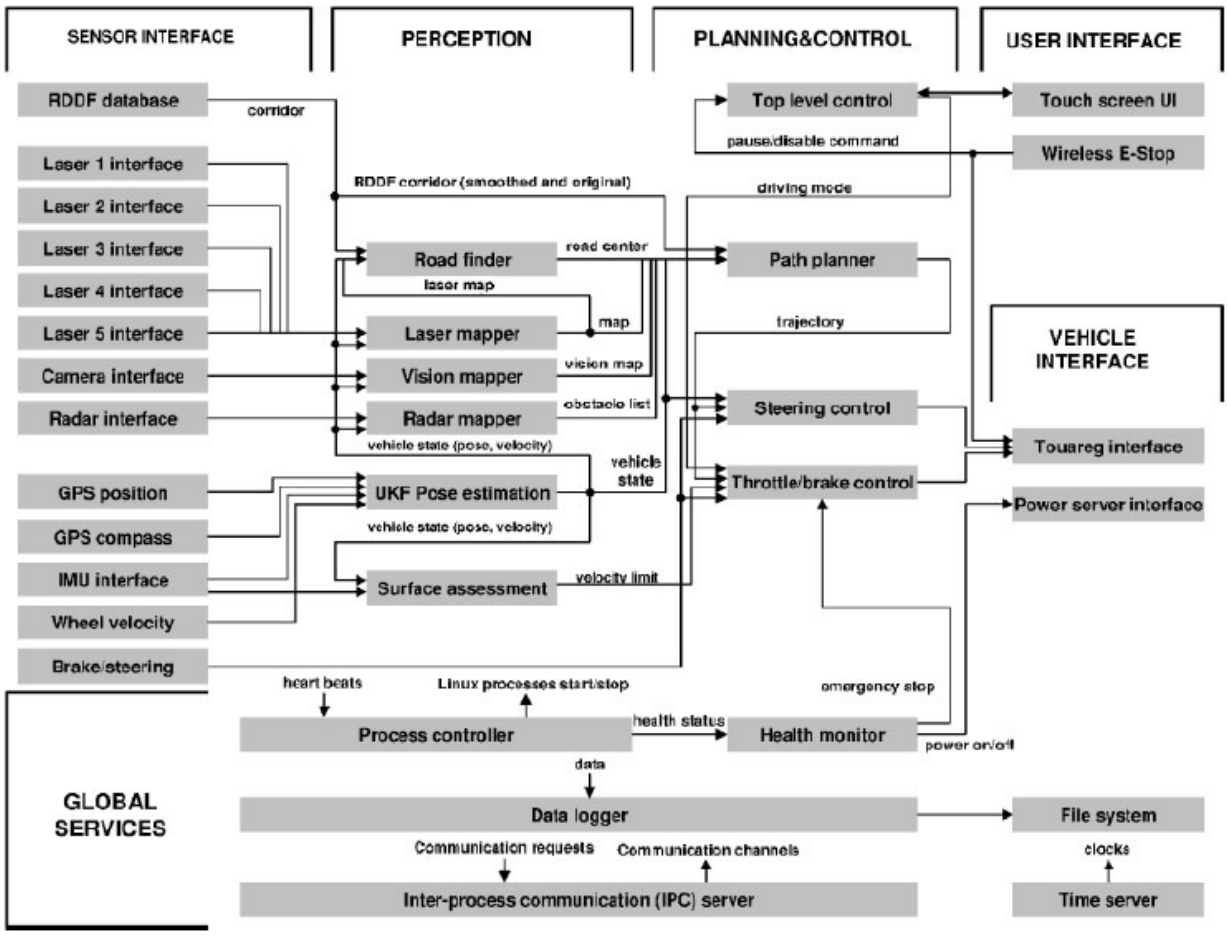
DARPA Challenge – Stanley

- Pierwsze miejsce w DARPA Grand Challenge w 2005 r.
- Stworzony na Stanford przez zespół Sebastiana Thruna (Google, Udacity)

DARPA Challenge – Stanley

- Pierwsze miejsce w DARPA Grand Challenge w 2005 r.
- Stworzony na Stanford przez zespół Sebastiana Thruna (Google, Udacity)
- Złożony system (oprogramowanie + sensory) zamontowane w standardowym VW Touareg R5 rozbudowanym o drive-by-wire

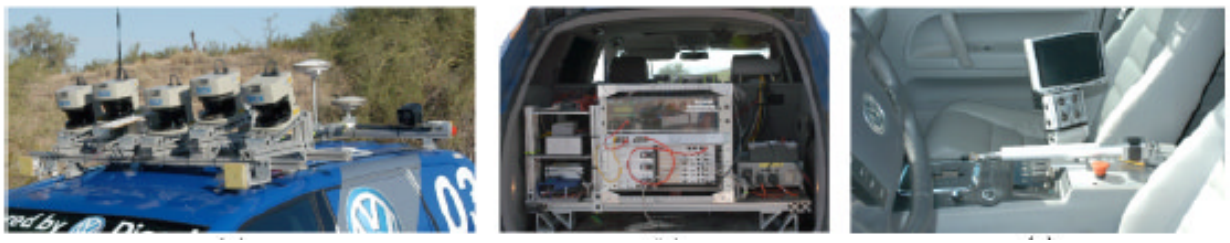
Software


SENSOR INTERFACE
PERCEPTION
PLANNING&CONTROL
USER INTERFACE
RDDF database
corridor
Laser 1 interface
Laser 2 interface
Laser 3 interface
Laser 4 interface
Laser 5 interface
Camera interface
Radar interface
GPS position
GPS compass
IMU interface
Wheel velocity
Brake/steering
RDDF corridor (smoothed and original)
Road finder
road center
laser map
Laser mapper
map
Vision mapper
vision map
Radar mapper
obstacle list
vehicle state (pose, velocity)
UKF Pose estimation
vehicle state
vehicle state (pose, velocity)
Surface assessment
velocity limit
Top level control
pause/disable command
driving mode
Path planner
trajectory
Steering control
Throttle/brake control
Touch screen UI
Wireless E-Stop
VEHICLE INTERFACE
Touareg interface
Power server interface
heart beats
Linux processes start/stop
emergency stop
Process controller
health status
Health monitor
power on/off
data
GLOBAL SERVICES
Data logger
File system
Communication requests
Communication channels
clocks
Inter-process communication (IPC) server
Time server

Hardware

- 6 serwerów (Pentium M) pod kontrola Linuxa

Hardware



- 6 serwerów (Pentium M) pod kontrola Linuxa
- 3 serwery dla oprogramowania, 1 do logowania, 2 zapasowe
  - 1 serwer odpowiedzialny za przetwarzanie sygnały wideo
  - 2 serwery odpowiedzialne za pozostałe funkcje (np. planowanie)

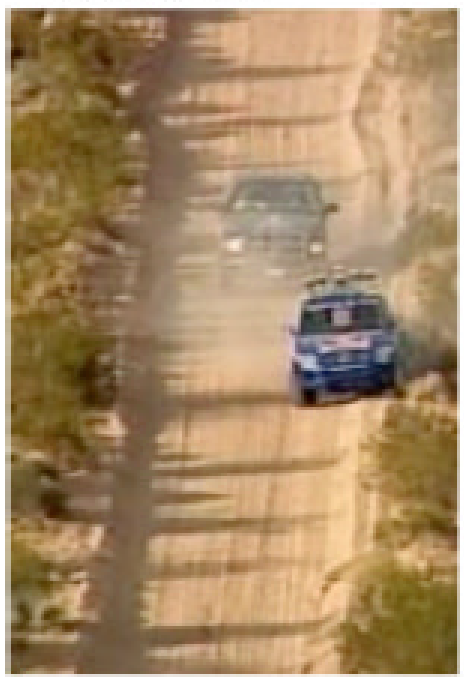
Planowanie

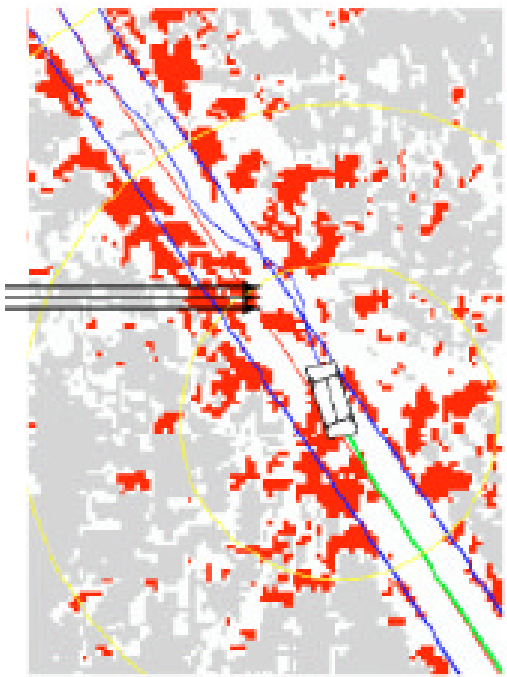
- Brak planowania długoterminowego
  - definicja trasy dostarczona przez organizatorów wyścigu
  - punkty kontrolne, szerokości korytarza, ograniczenia prędkości

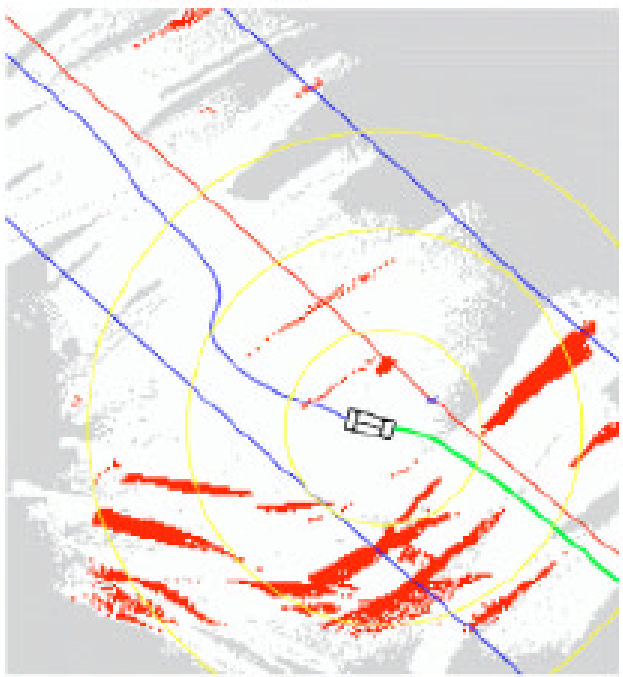
Planowanie

- Brak planowania długoterminowego
  - definicja trasy dostarczona przez organizatorów wyścigu
  - punkty kontrolne, szerokości korytarza, ograniczenia prędkości
- Planowanie krótkoterminowe (przeszukiwanie)
  - poruszanie się blisko środka korytarza
  - omijanie przeszkód (np. innych pojazdów)
  - maksymalizacja prędkości (w ramach ograniczeń) i minimalizacja czasu przejazdu

Planowanie







DARPA Challenge – Junior

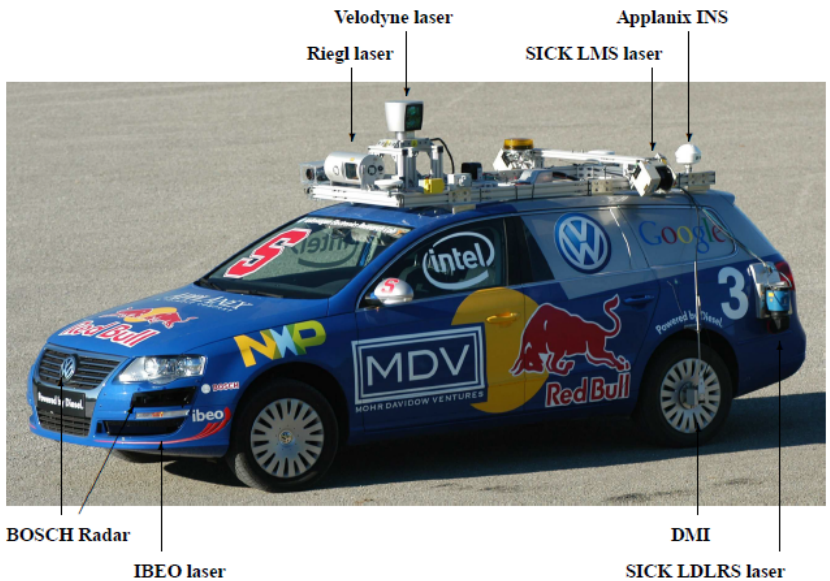
- Drugie miejsce w DARPA Grand Challenge w 2007 r.

DARPA Challenge – Junior

- Drugie miejsce w DARPA Grand Challenge w 2007 r.
- Przygotowany przez zespół, który wcześniej opracował Stanley'a

DARPA Challenge – Junior

- Drugie miejsce w DARPA Grand Challenge w 2007 r.
- Przygotowany przez zespół, który wcześniej opracował Stanley'a
- System zamontowany w VW Passat Wagon (z systemem *drive-by-wire*)



Organizacja zawodów

- Dokładna mapa otoczenia (drogi, pasy ruchu, znaki stopu, parkingi, punkty kontrolne)

Organizacja zawodów

- Dokładna mapa otoczenia (drogi, pasy ruchu, znaki stopu, parkingi, punkty kontrolne)
- Dodatkowa mapa lotnicza okolicy – możliwość doprecyzowania mapy przez organizatorów

Organizacja zawodów

- Dokładna mapa otoczenia (drogi, pasy ruchu, znaki stopu, parkingi, punkty kontrolne)
- Dodatkowa mapa lotnicza okolicy – możliwość doprecyzowania mapy przez organizatorów
- Bez świateł na skrzyżowaniach i dodatkowych znaków drogowych (poza mapą)

Organizacja zawodów

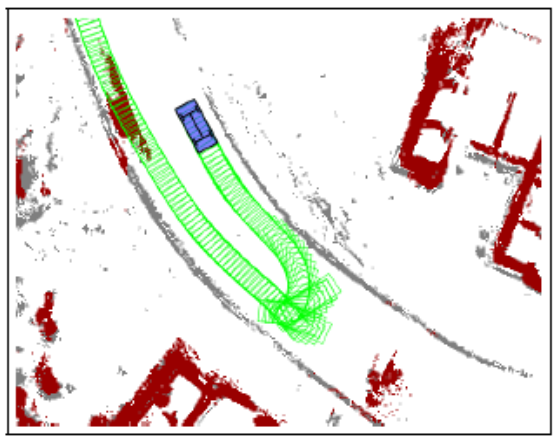
- Dokładna mapa otoczenia (drogi, pasy ruchu, znaki stopu, parkingi, punkty kontrolne)
- Dodatkowa mapa lotnicza okolicy – możliwość doprecyzowania mapy przez organizatorów
- Bez świateł na skrzyżowaniach i dodatkowych znaków drogowych (poza mapą)
- 3 misje do wykonania, podczas których trzeba było unikać kolizji, przestrzegać przepisów ruchu drogowego i minimalizować czas realizacji

Software

- Architektura warstwowa taka sama, jak w przypadku Stanley'a

Software

- Architektura warstwowa taka sama, jak w przypadku Stanley'a
- Rozbudowana część związana z planowaniem – dwa moduły do planowania ruchu na trasie i na parkingu (mniej standardowe manewry)



Hardware

Dwa serwery (*quad core*) działające pod kontrolą Linuxa