**METAREASONING USING ANALOGY AND ITS APPLICATIONS TO ROBOTICS**

Adrián Lendínez Ibáñez 1400881

Supervisor: Doctor Renxi Qiu

Contents

[Problem definition: 2](#_Toc129010595)

[Research questions: 2](#_Toc129010596)

[Why experiences? What are they formally? – QValues 2](#_Toc129010597)

[Memory bank of experiences: 2](#_Toc129010598)

[Experience 1: Known situation 2](#_Toc129010599)

[Experience 2: Known situation 3](#_Toc129010600)

[Experience 3: Unknown situation 3](#_Toc129010601)

[Experience 4: Unknown situation 3](#_Toc129010602)

[Iterative estimation of experiences believes: 4](#_Toc129010603)

[Definition of MDP for choosing the right experience: 5](#_Toc129010604)

[Analogy calculation: 6](#_Toc129010605)

[State compression problem: 6](#_Toc129010606)

[Find similarity of states: 6](#_Toc129010607)

# Problem definition:

Problem is: How to reuse previous experiences knowledge in a new/semi new situation using analogy?

**Assuming**: some historical data available at the time = 0. E1 & E2.

Robot does not know if it is in E1 or E2.

Verify if making good analogy, how fast convergence to the right experience (validation of theory).

 (If analogies are wrong, it should take more time to converge.)

# Research questions:

\* If newSittuation is either E1 or E2, this meta reasoning using analogy is always better than random just my pure logic. Prove by using simulation and many experiments averaging them.

\* The real interesting problem is if newSittuation is not E1 or E2 but resemblances somewhat to E1 or E2.

\* If newSittuation is a linear combination of E1 and E2, can the algorithm realize this and make better informed decision and create new E3?

\* **Big question:** what is the best way to get best analogy at the beginning?

# Why experiences? What are they formally? – QValues

Experiences bundle together states.

# Memory bank of experiences:

The collection of experiences the robot already has.

## Experience 1: Known situation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R | 10 | -100 | -100 | 5 | D100 |
| 10 | 15 | -100 | -100 | 5 | 80 |
| 10 | 15 | 15 | 15 | 15 | 60 |
| 0 | 18 | 20 | 30 | 50 | 55 |
| -10 | -10 | 10 | 10 | 34 | 45 |

All cells have a 100% lighting condition.

Experience1 90%  (Action success 95%)     Localisation (100%)

Action success: calculated by running the experiment many times and averaging.

## Experience 2: Known situation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R | 10 | -100 | -100 | 0 | D100 |
| 10 | 15 | -100 | -100 | 0 | 80 |
| 10 | 10 | 0 | 0 | 0 | 60 |
| 0 | 18 | 20 | 30 | 50 | 55 |
| -10 | -10 | 10 | 10 | 34 | 45 |

All cells have a 0% lighting condition.

Experience2 10% (Action success 65%)   Localisation (100%)

Action success: calculated by running the experiment many times and averaging.

## Experience 3: Unknown situation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X (state x) | y |  |  |  |  |
| z |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Half the map (left hand side) has 0% lighting condition.

Experience3 10% (Action success 65%)   Localisation (100%)

Action success: calculated by running the experiment many times and averaging.

## Experience 4: Unknown situation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

(left hand side) has 0% lighting condition. Middle of map, 50% lighting condition and right 100%.

Experience4 10% (Action success 65%)   Localisation (100%)

Action success: calculated by running the experiment many times and averaging.

# Iterative estimation of experiences believes:

Initial data:

E1= Believe in Exp1 initial analogy P[E1(0)]

Under E1, the success rate of the Action A is = 95%

E2= Believe in Exp2 initial analogy P[E2(0))

Under E2, the success rate of the Action A is = 65%

A=Observed action success rate, to be collected during the iterations

Smoothing of the observed success rate using NLP: <https://en.wikipedia.org/wiki/Additive_smoothing>

Text

Description automatically generated

1st iteration

Given observed action success rate as

The believe of the Exp1 is updated as

New believes

The believe of the Exp2 is updated as

New believes

2nd iteration

Given observed action success rate as

The believe of the Exp1 is updated as

New believes

The believe of the Exp2 is updated as

New believes

Ith iteration

Given observed action success rate as

The believe of the Exp1 is updated as

New believes

The believe of the Exp2 is updated as

New believes

# Definition of MDP for choosing the right experience:

**Need to be defined: MDP\_reward for choosing right experience.**

**Strategies:**

-Change believe of E1 to E2 and act according to their policies dynamically or select a policy based on initial analogy and stick with it until the end.

State[0] = [map, robotPose, robot\_localization\_state]

E1 = Q\_table\_1

E2 = Q\_table\_2

Def Analogy(State[0], Experience):

Return analogical\_distance 🡪 P[E1(0)]

Changing experience reward -1000000000

Iteration 0:

Given State[0], E1 and E2.

P[E1(0)] = Analogy(State[0], E1) [Probab of E1 in t=0]

Historical + current state + Current\_confidence

P[E2(0)] = Analogy(State[0], E2) [Probab of E2 in t=0]

= MDP E1 [Prob of success of action in context E1] We mean prob of doing exactly the action even with noise. To calculate:

\* E1 = state1,state2,state3…stateN

\*P(MoveUp1|state) = (P(MoveUp1| state1) + P(MoveUp1| state2) + P(MoveUp1| stateN)) % N

\*P(MoveDown1|state) = (P(MoveDown1| state1) + P(MoveDown1| state2) + P(MoveDown1| stateN)) % N

\* P(Action| Ex1) = P(MoveUp1|state) + P(MoveDown1|state) + P(MoveLeft1|state) + …

= General given value. [Prob of success of action in any context] To calculate:

If the number of cells and parameters such us lighint are fixed, it cab be calculated accurately. Do we assume this?

(P(Action| Ex1) + P(Action| Ex2) + P(Action| ExN)) % N

Iteration 1:

Given State[1], E1 and E2.

=

Implement randomness in the terrain: smooth, raft.

Smooth map: expected 90% of terrain to be smooth.

Raft map: expected 90% raft.

# Analogy calculation:

Analogy is used to guess the most similar previous experience given some observations and initial state.

State compression problem: efficient way of representing state. Absolute position (fast). (Memory of humans)

Find similarity of states: how to compare similarity. If similar reward it may be the same situation. Estimate reward and have some estimation. If reward is low, maybe not do so much calculation. If the other way around.

Scenarios:

PMDP: states are infinite. You can assume saying there are limited states.

**How much tolerance the robot has? Depending on the task ahead.**

How to do the Verification?

Unknown Situation 1   (Action success ??%):

Skill == expiernces.

How to select skill:

Updating confidence based on fuzzy reasoning.

S(1,2,3)

S\_policy(1,2,x,y,5) 2 reward -1

Unknown Situation 2

Calculate Q-table using Bellman equation and epsilon greedy algorithm

1. Define formually in maths the 1+1

Decompose and recombine experience (human). Verify and experience repeat.

E = Q table

action1, action2, action3

State 1 : 1 2 2.3

-Table 3

-Circle detection & MDP.

1. First step: use historical data and safety constraints to decide

E1: Prob(70)% [Confidence: 30%]

E2: Pro(50) % [Confidence: 30%]

E3: 25% [Confidence: 30%]

For each believe in exp we should have a confidence level.

MDP Task completion: Reward for completing task:

MDP for building up world model: Reward for maximizing confidence:

To switch: safety + confidence + probability. => reward for switching -1000

Confidence of none of the above: new E4

Assumptions: world is static for now. Static distribution of randomness (Smoothness of terrain of each cell).

\* Action points:

1. Create Exp1 Smooth terrain. 🡪 generate Q table Ex1 Skill Smooth terrain.
2. Create Exp Raft terrain .
3. Calculation P[E1(0)] of Prob Exp with Analogy: Historical data + current state + Current\_confidence
4. Implement Iterative estimation of experiences believes (Bayes equations)
5. MDP for building up world model.
   1. Reward for choosing experiences for each action step.
   2. Action: Switch or not switch.
   3. States: Prob high/low + conf high/low

Table for update iteration of MDP.

1. For exp.
2. Confidence.

\* Evaluation:

-Given a policy of 90% smooth and another one with 10% smooth. Select the best policy for 75% of smoothness using the 2 MDPs. (By assumption 75% is closer to 100% meaning First policy should do better.) Need to evaluate properly.