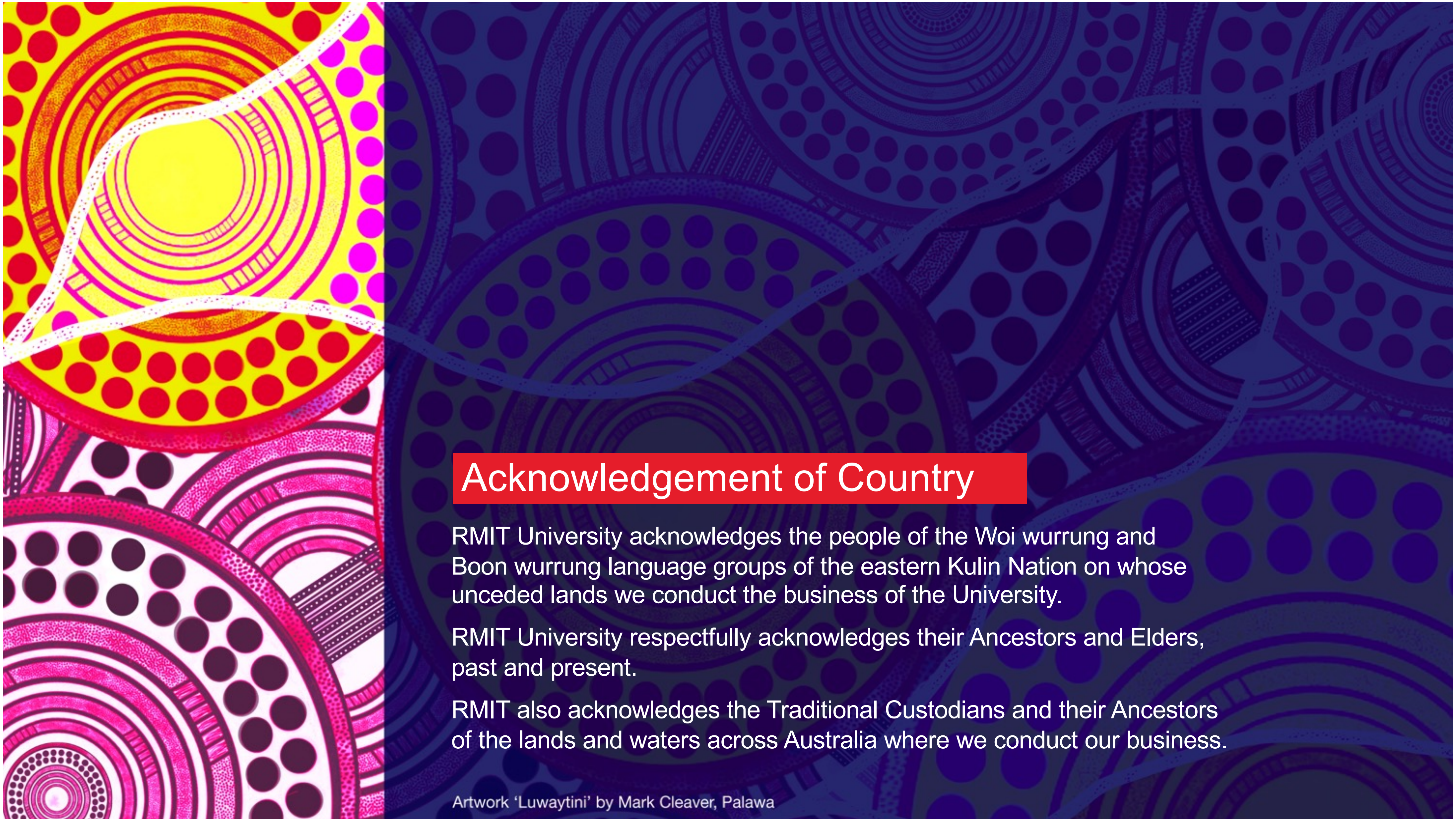
Practical Applications of Artificial Intelligence for Robotics

Day 5: Task Planning

Dr. Timothy Wiley
School of Computing Technologies
RMIT University

ESSAI July 2023

RMIT UNIVERSITY



Acknowledgement of Country

RMIT University acknowledges the people of the Woi wurrung and Boon wurrung language groups of the eastern Kulin Nation on whose unceded lands we conduct the business of the University.

RMIT University respectfully acknowledges their Ancestors and Elders, past and present.

RMIT also acknowledges the Traditional Custodians and their Ancestors of the lands and waters across Australia where we conduct our business.

Artwork ‘Luwaytini’ by Mark Cleaver, Palawa

Motivation

ESSAI July 2023

Find a Kitchen Item

Consider the following problem for a service robot:

- *“Human says to robot: Go a fetch a can of <beverage> from the fridge and bring it to me in the lounge room"*

Questions:

- What type of information do we need to know for this problem?
- What type of information should be represented at the Deliberative layer?
- What type of "deliberations" need to be made for planning?

An Observation

The summer school have a variety of courses on:

- Knowledge Representation and Reasoning
- Symbolic Logics
- Symbolic Planning

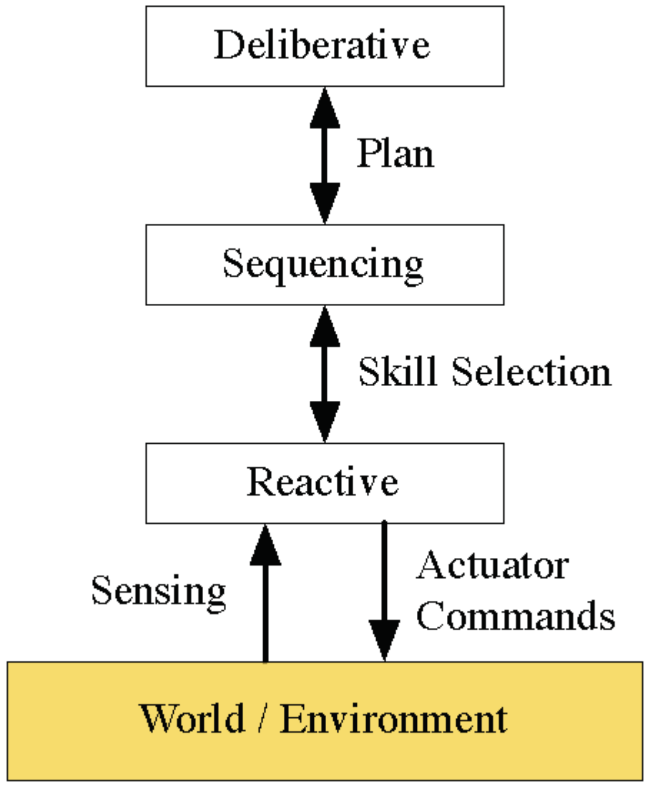
What will be considered in this course is the *practical* issues of deploying a symbolic planner onto a robotic platform

Software Architectures

Recap

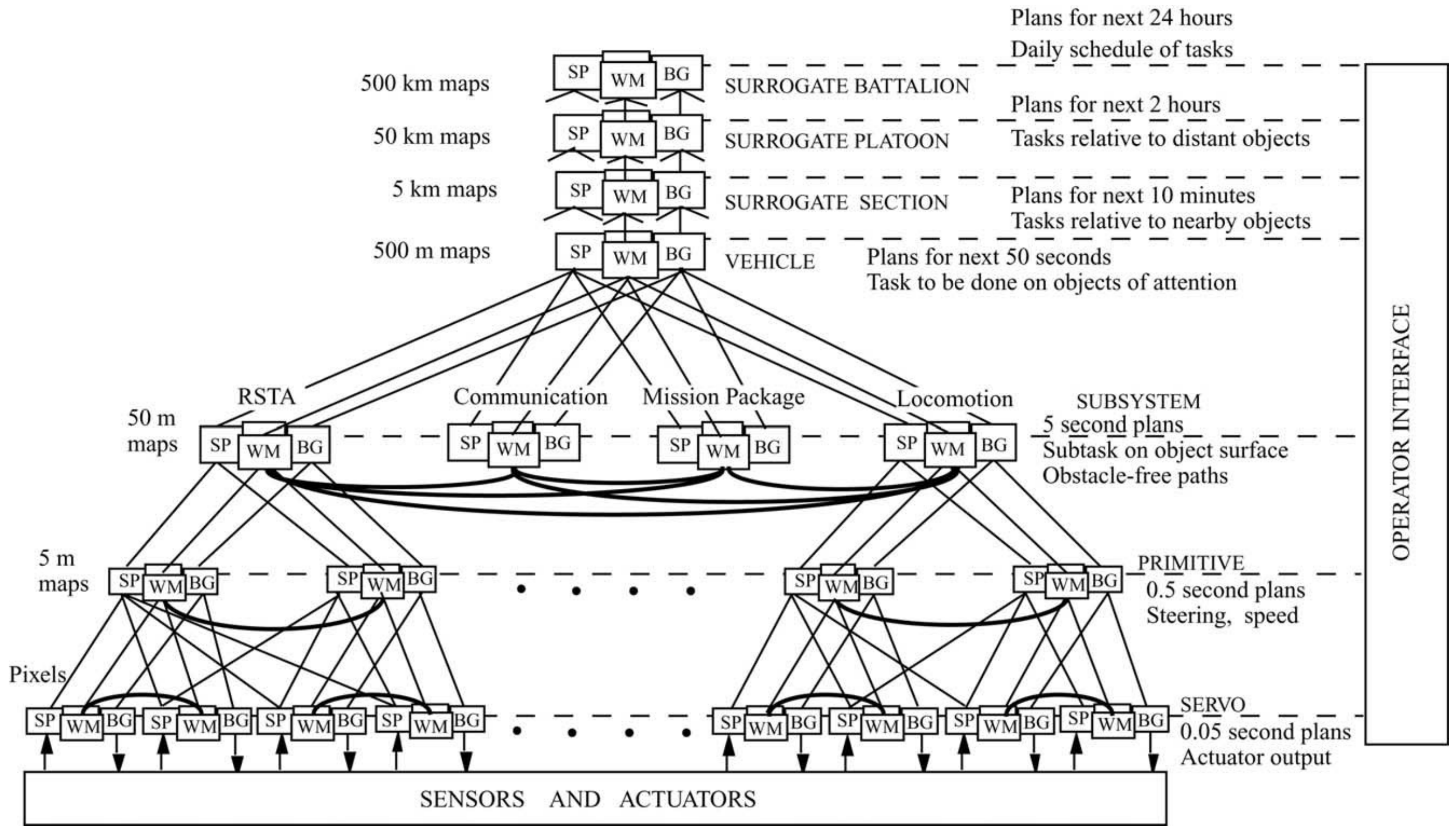
ESSAI July 2023

Software Architectures: Three-Layer



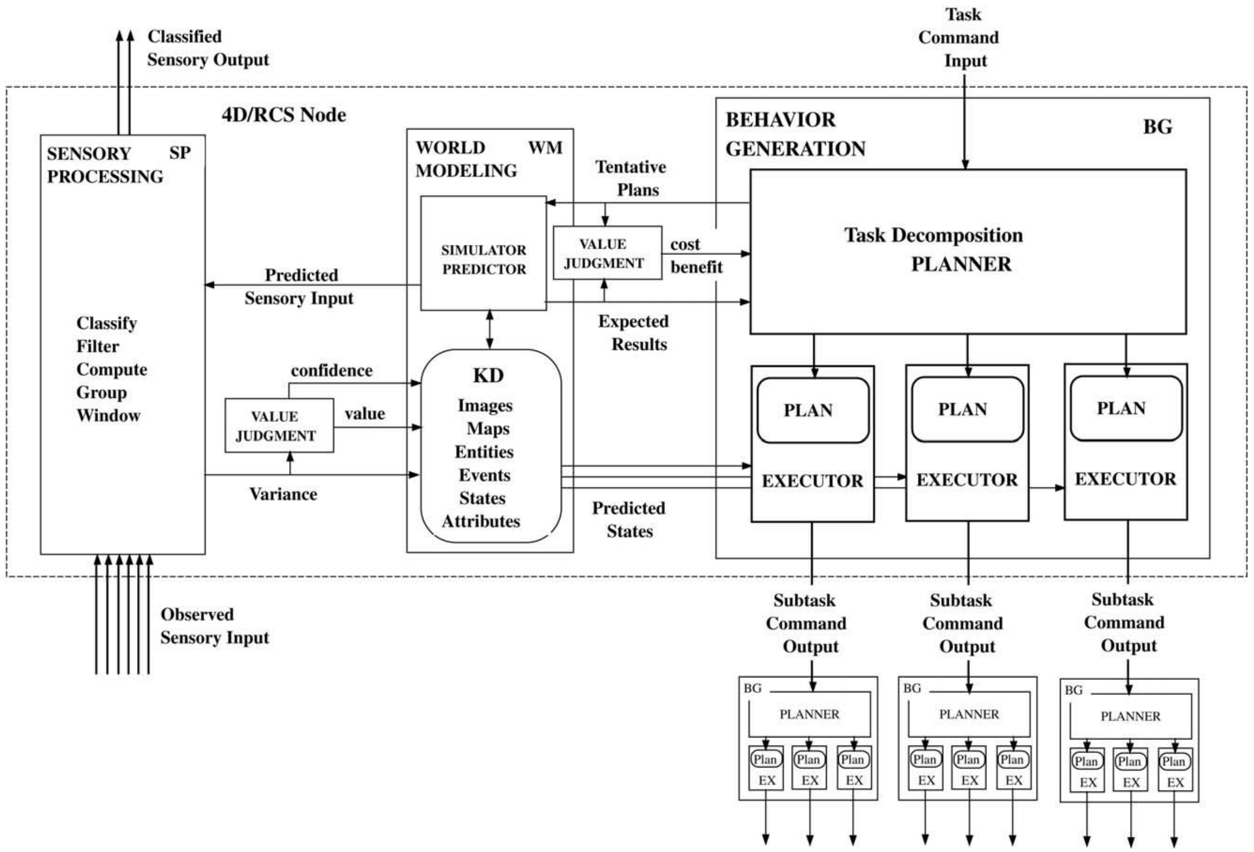
Bonasso, P. et. al. (1997). Experiences with an architecture for intelligent, reactive agents. Journal of Experimental & Theoretical Artificial Intelligence 9(2-3):237– 256

Software Architectures: RCS



Albus, J. S. & Barbera, A. J. (2005) RCS: A cognitive architecture for intelligent multi-agent systems. Annual Rev Control 29, 87–99.

Software Architectures: RCS



Albus, J. S. & Barbera, A. J. (2005) RCS: A cognitive architecture for intelligent multi-agent systems. Annual Rev Control 29, 87–99.

Symbolic Task Planning

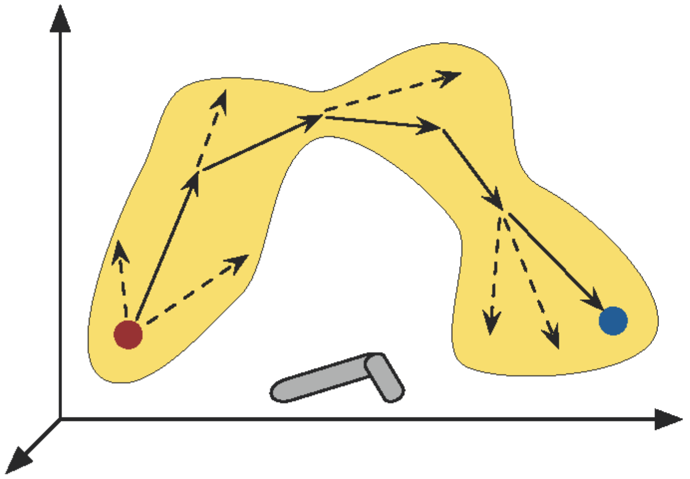
For robotics

ESSAI July 2023

Task Planning formulation

We the discussions here we are going to define symbolic task planning as a symbolic state-action plan, with:

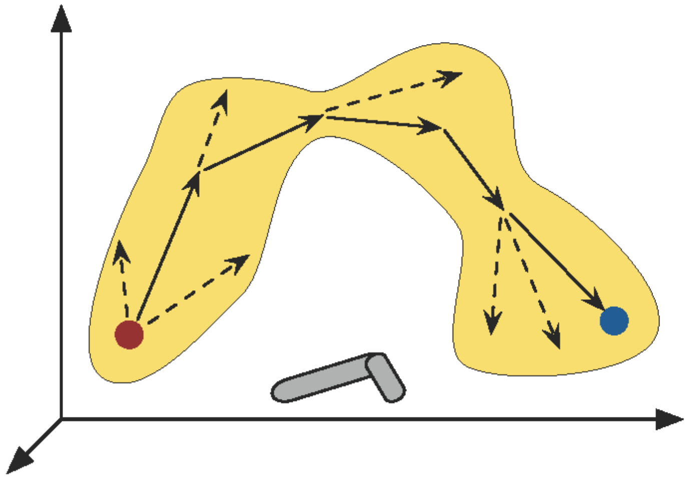
- A State, s, that are symbolic representations of the numeric state space of the robot, *and*, the environment
- An Action, a, are symbolic representations of robot movement.
  - Note that actions do not necessarily need to directly correlate to one actuator movement
- Actions are temporal, and may take a variable length of time to execute



Task Planning formulation

Therefore a *task plan* is:

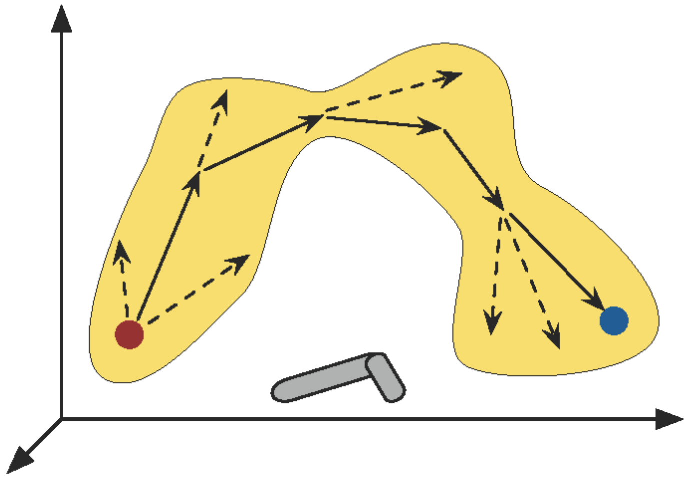
- A sequence of actions that, when executed, enable the robot so change from an initial state to a goals state.
- Actions connect intermediate states



Executing Task Plans

A task plan is (naively) executed by:

1. Executing the first action of the task plan until the first intermediate state is reached
2. Execution continues with subsequent actions, until the relevant intermediate states are reached
3. Execution terminates on reaching the goal state.



STRIPS

ESSAI July 2023

STRIPS Planning (as done on Shakey!)

STRIPS Planning is a very simple method for symbolic planning.

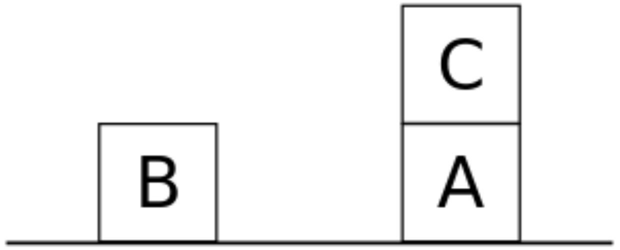
- State: Represented by logic predicates
- Actions:
  - Name
  - Precondition
  - Postcondition
  - Add list
  - Delete list

A note for ESSAI’23

It’s likely many are quite familiar with STRIPS planning, and far more complex planning languages.

We are going to start with STRIPS to investigate the issues of the practical side of task planning on robot systems, and motivate more modern planning languages.

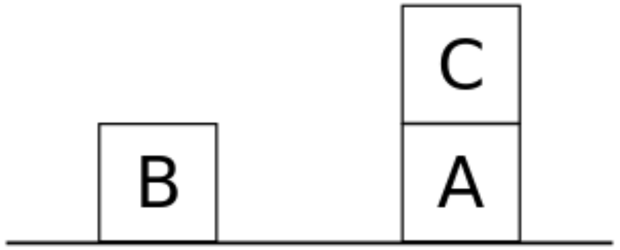
Blocks World



States:
- On(A,B)
- Clear(A)
- OnTable(A)
- Holding(A) ← For robot actions!

Actions:
- Pickup(X)
- Putdown(X)
- Stack (X,Y)
- Unstack (X,Y)

Blocks World



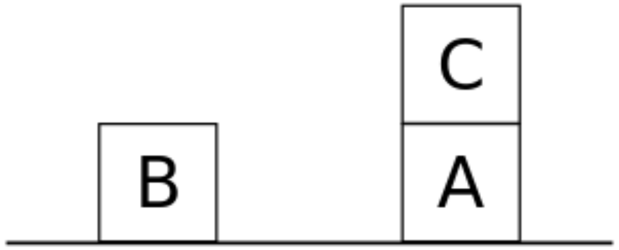
Pickup(x):

- Preconditions:
  - Clear(x)
  - OnTable(x)
  - Holding(∅)
- Delete list:
  - Clear(x)
  - OnTable(x)
  - Holding(∅)
- Add list:
  - Holding(x)

Putdown(x)

- Precondition:
  - Holding(x)
- Delete list:
  - Holding(x)
- Add list:
  - Clear(x)
  - OnTable(x)
  - Holding(∅)

Blocks World



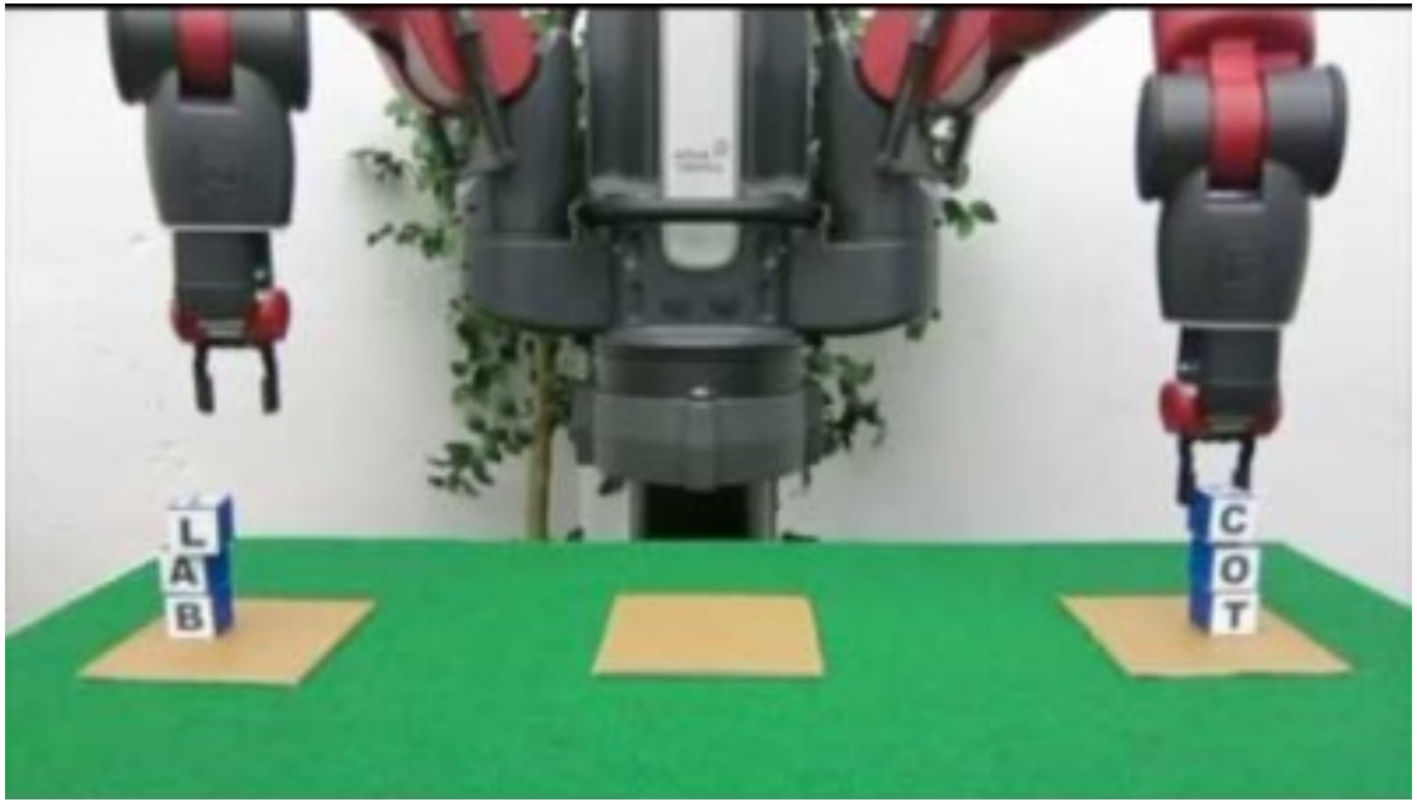
Unstack(x,y):

- Preconditions:
  - Clear(x)
  - On(x,y)
  - Holding(∅)
- Delete list:
  - Clear(x)
  - On(x,y)
  - Holding(∅)
- Add list:
  - Holding(x)
  - Clear(y)

Stack(x,y)

- Precondition:
  - Holding(x)
  - Clear(y)
- Delete list:
  - Holding(x)
  - Clear(y)
- Add list:
  - Clear(x)
  - On(x,y)
  - Holding(∅)

Baxter w/ Blocks World (Not quite STRIPS)



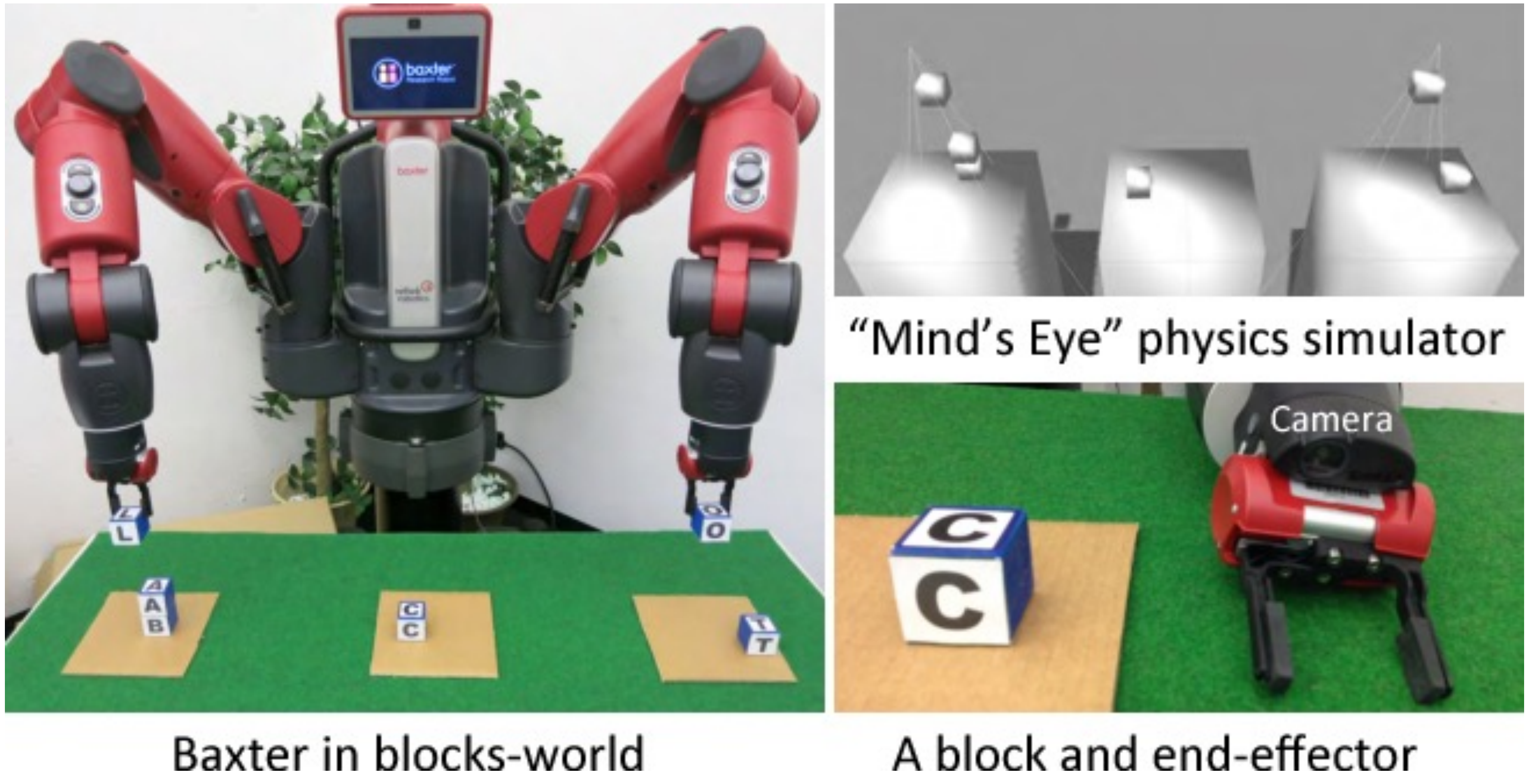
B. Hengst, et. al, A framework for integrating symbolic and sub-symbolic representations. 25th International Joint Conference on Artificial Intelligence IJCAI-16. New York, New York, USA, 2016.

Issues that STRIPS highlights

STRIPS Planning is useful for highlighting challenges for symbolic planning in real-world robotics:

- “Perfect” state assumption
- Symbolic to sub-symbolic correlation
- Observability / Occlusion
- Durative Actions
- Actions failure
- Outside Influences
- Response to real-time sensor updates
- Parallel Action

Baxter w/ Blocks World (Not quite STRIPS)



B. Hengst, et. al, A framework for integrating symbolic and sub-symbolic representations. 25th International Joint Conference on Artificial Intelligence IJCAI-16. New York, New York, USA, 2016.

Symbolic to Sub-Symbolic

GOLOG

ESSAI July 2023

State Representation at TLA layers

In a deliberative layer a symbolic representation, such as in blocks-world, may represent the world state by:

- OnTable(A)
- On(B,A)

Below the reactive layer is sub-symbolic representations, that "does not have symbolic entities or discrete elements" but, typically, involves some form of numeric information.

In the reactive layer, this sub-symbolic information are sensor information, including the camera. For the block's world this would be a camera image of the layout of the world

State Representation at TLA layers

An current open research challenge is how to "bridge" these representations in the sequencing layer. This is because, the symbolic world is "idealistic", but the "real-world" has:

- Noise
- Error
- Non-determinism
- Hidden information
- Dynamic changing information

Potential Solutions

There are many potential solutions:

- Improve the "richness" of symbolic representations - that is, the simple modelling is insufficient
- Embed sub-symbolic information in symbolic representations, such as:
  - OnTable (A, x, y, z)
  - On(C,A, <relative position)

Using symbolic modelling of continuous variables, rather than pure symbolic terms

Behaviour Trees

ESSAI July 2023

Behaviour Tree

Behaviour trees are a type of decision tree that allow:

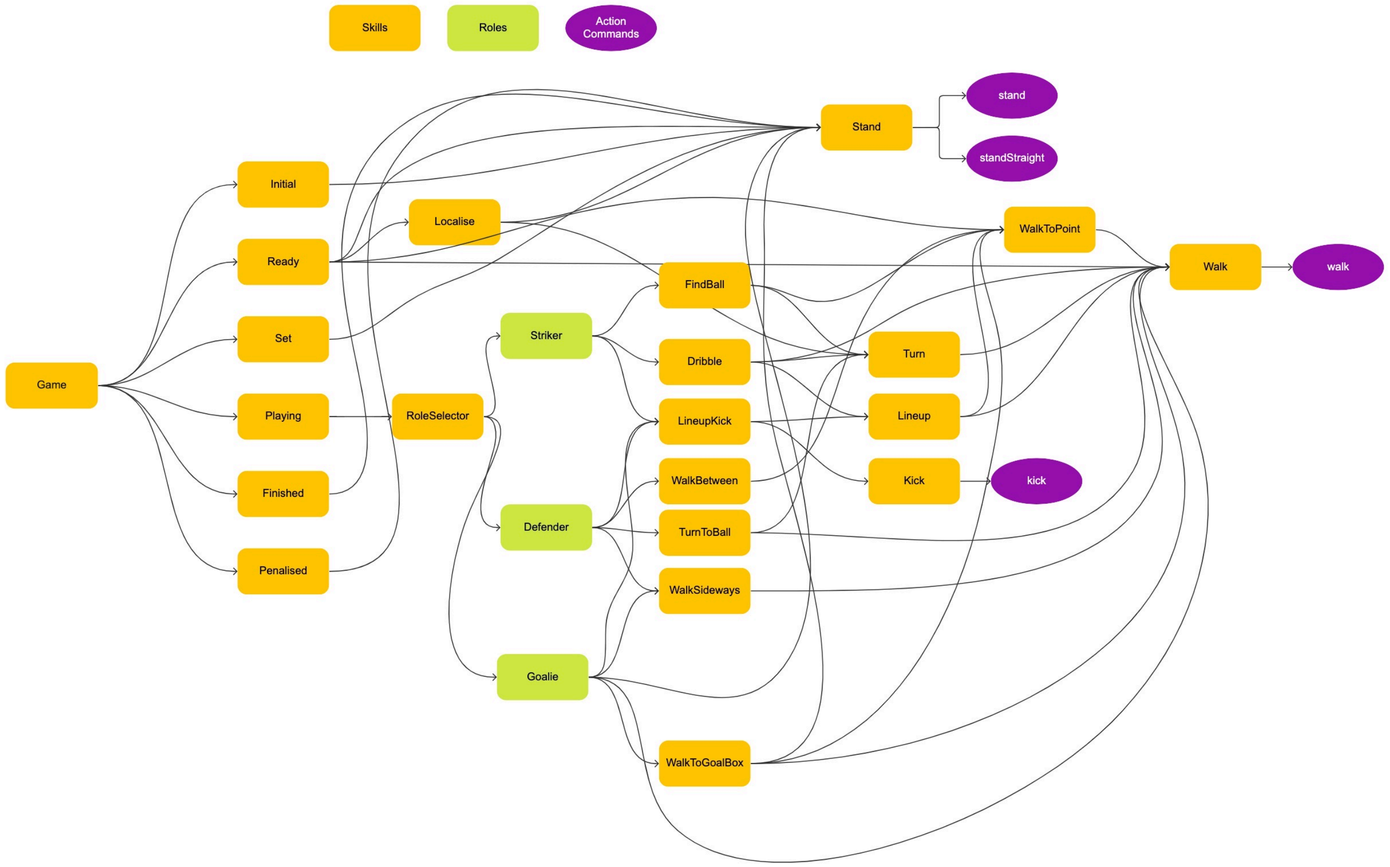
- Modularisation of skills
- Quick evaluation for planning / skill-selection
- Durative Actions
- Parallel Actions

Nao Behaviour Tree

The RMIT RedbackBots (based on rUNSWift) independently control:

- Head
- Body

The Body behaviour tree is shown



Behaviour Tree Issues

Issues with Behaviour Trees include:

- Complexity of tree structure
- No explicit “planning”. Instead planning is a consequence of the tree skill-selection
- High-level decision making can drastically swap behaviours

Agent Programming Languages

GOLOG

ESSAI July 2023

GOLOG Programming Language

Golog is a logic-based agent programming language built on-top of Prolog, that facilitates the representation and execution of actions:

1. Actions: Predicates that can be any operation that modifies the state of the environment or the agent itself.
2. Fluents: represent state information that can change over time. This allows reasoning about dynamic aspects of the environment and update the state information.
3. Concurrent Actions: execution of multiple actions concurrently.
4. Non-Deterministic Choice: This allows the agent to determine different possible actions where it is unknown the probably of either alternative to explore different possibilities.

Goals: represent the desired state of the agent

Task Planning Problem

Devise a task plan to *“go and get a beverage from the fridge”.*

Assumes knowledge of:

- Rooms of the house
- Objects in each room
- Relationship between objects
- Tracking of “person” locations
- Tracking state of “modifiable” objects

GOLOG Program: Fluents

```
:- dynamic at/1, opened/1, has/1.   % Declare dynamic predicates for fluents
```

GOLOG Program: Actions

```
% Move between locations
move(Location1, Location2) :-
    can_move(Location1, Location2),
    assert(at(Location2)),
    retract(at(Location1)).

% Open the fridge
open(Fridge) :-
    at(Location),
    can_open(Location, Fridge),
    assert(opened(Fridge)).

% Close the fridge
close(Fridge) :-
    at(Location),
    opened(Fridge),
    can_close(Location, Fridge),
    retract(opened(Fridge)).
```

```
% Grab an item from the fridge
grab(Item) :-
    at(Location),
    opened(Fridge),
    can_grab(Location, Item),
    assert(has(Item)).
```

GOLOG Program: Rules

```
% Define the conditions for allowed moves
can_move(Location1, Location2) :-
    connected(Location1, Location2).

% Define the conditions for opening the fridge
can_open(Location, Fridge) :-
    fridge(Location, Fridge),
    \+ opened(Fridge).

% Define the conditions for closing the fridge
can_close(Location, Fridge) :-
    fridge(Location, Fridge),
    opened(Fridge).

% Define the conditions for grabbing an item from the fridge
can_grab(Location, Item) :-
    fridge(Location, fridge),
    item_in_fridge(fridge, Item).
```

GOLOG Program: Goal

```
% The goal is to have a can of coke
goal :-
    has(coke).
```

GOLOG Program: Procedures

```
% Procedure to navigate to a specific location
goto(Location) :-
    at(CurrentLocation),
    move(CurrentLocation, Location).

% Procedure to open the fridge and grab an item
open_and_grab(Item) :-
    open(fridge),
    grab(Item),
    close(fridge).

% Procedure to choose a fridge non-deterministically
choose_fridge(Fridge) :-
    (fridge(kitchen, Fridge) ; fridge(garage, Fridge)).
```

GOLOG Program: Main planning program

```
main :-
    goal,   % Define the goal
    goto(kitchen),
    choose_fridge(Fridge),
    open_and_grab(coke).

% Initial State
at(starting_location).
connected(starting_location, kitchen).
connected(kitchen, garage).
fridge(kitchen, fridge).
fridge(garage, fridge).
item_in_fridge(fridge, coke).
```

The End

Dr. Timothy Wiley
School of Computing Technologies
RMIT University

ESSAI July 2023

RMIT UNIVERSITY

Thank you

I would like to express my deep thanks for everyone who has attended these classes over the week. I have very much enjoyed this week. I hope that I have provided some insight into what robotics can offer and how to apply AI in real-world settings.

Please feel free to connect and stay in touch!

- Course Resources: https://timothy-wiley.github.io/essai.html
- LinkedIn: https://www.linkedin.com/in/timothy-wiley-948a9113/
- Email: timothy.wiley@rmit.edu.au
- Twitter: https://twitter.com/time_wiley

Noon Gudgin

Thank you

ESSAI July 2023