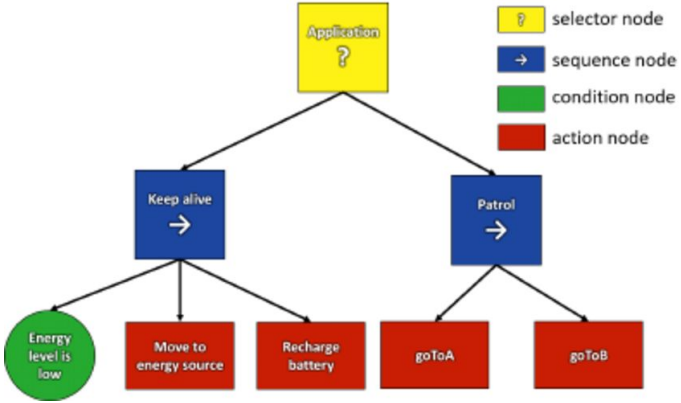
無人機智慧系統開發與實作

Behavior Tree 介紹

Behavior Tree

- A mathematical model of plan exection

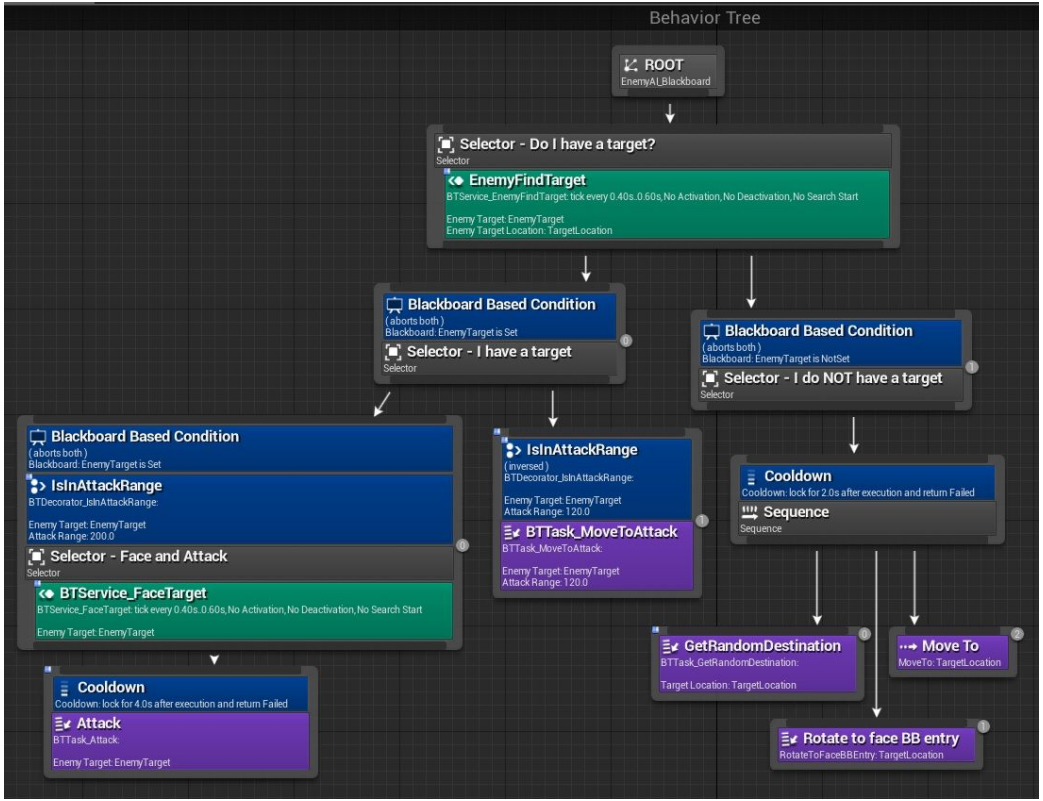
- Robotics: *Behavior Trees in Robotics and AI: An Introduction*
- Mission management: *Behavior Trees for UAV Mission Management*
- Games: Mario, *Learning of behavior trees for autonomous agents*
- Medical: *Semi-autonomous simulated brain tumor ablation with ravenii surgical robot using behavior tree*



https://en.wikipedia.org/wiki/Behavior_tree_(artificial_intelligence,_robotics_and_control)
de Lucca Siqueira, F., & De Pieri, E. R. (2015). A context-aware approach to the navigation of mobile robots. In *Proceedings on Simposio Brasileiro de Automacao Inteligente. XII SBAI* (pp. 1-6).

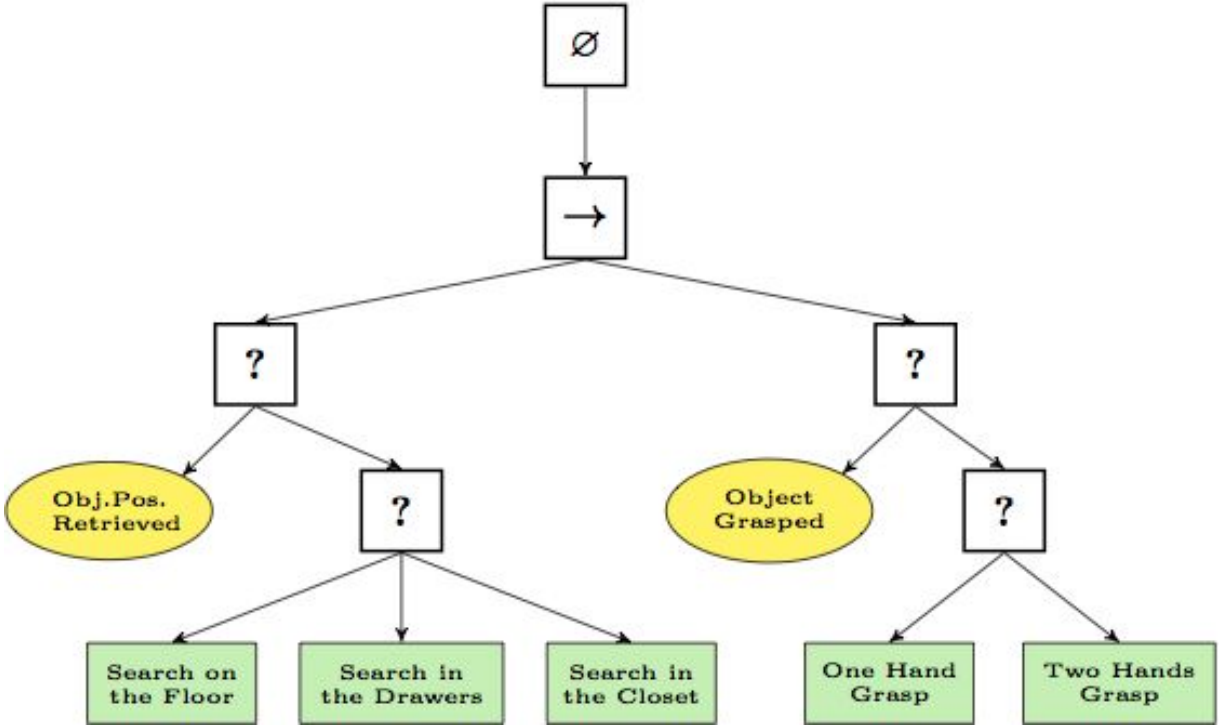
Behavior Tree: Background

- Developed in computer game industry
  - Control Non-Player Characters(NPC)
- why BT?
  - Modularity
    - enables reuse of code
    - incremental design of functionality
    - efficient testing
- Modularity > Robot



Behavior Tree: concepts

- A directed tree
  - root
  - control flow nodes
    - sequence node
    - fallback node
    - parallel node
  - execution nodes
    - action node
    - condition node
    - decorator node
  - node status
    - succeed
    - fail
    - running



Behavior Tree: control flow nodes

- Sequence node:

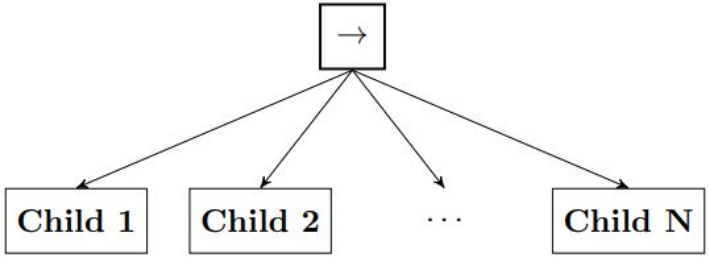


Fig. 1.2: Graphical representation of a Sequence node with N children.

Algorithm 1: Pseudocode of a Sequence node with N children

```
for i ← 1 to N do
    childStatus ← Tick (child(i))
    if childStatus = Running then
        return Running
    else if childStatus = Failure then
        return Failure
return Success
```

Behavior Tree: control flow nodes

- Fallback(selector) node:

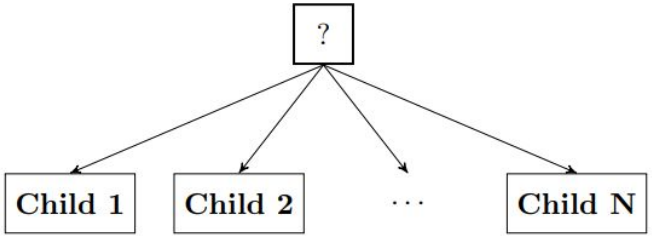


Fig. 1.3: Graphical representation of a Fallback node with N children.

Algorithm 2: Pseudocode of a Fallback node with N children

```
for i ← 1 to N do
    childStatus ← Tick (child(i))
    if childStatus = Running then
        return Running
    else if childStatus = Success then
        return Success
return Failure
```

Behavior Tree: control flow nodes

- Parallel node:

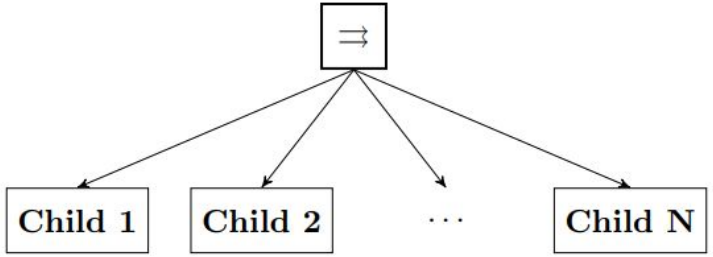
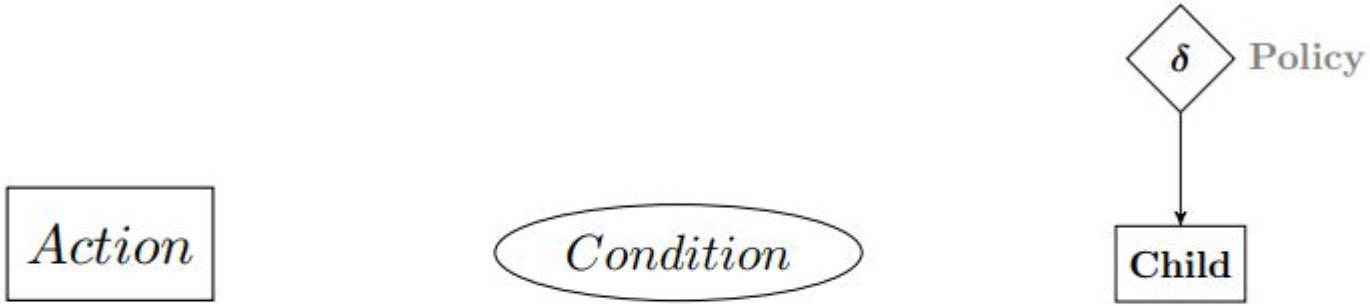


Fig. 1.4: Graphical representation of a Parallel node with N children.

Algorithm 3: Pseudocode of a Parallel node with N children and success threshold M

```
for i ← 1 to N do
    childStatus(i) ← Tick (child(i))
if Σ_{i:childStatus(i)=Success} 1 ≥ M then
    return Success
else if Σ_{i:childStatus(i)=Failure} 1 > N − M then
    return Failure
return Running
```

Behavior Tree: execution nodes



(a) Action node. The label describes the action performed.

(b) Condition node. The label describes the condition verified.

(c) Decorator node. The label describes the user defined policy.

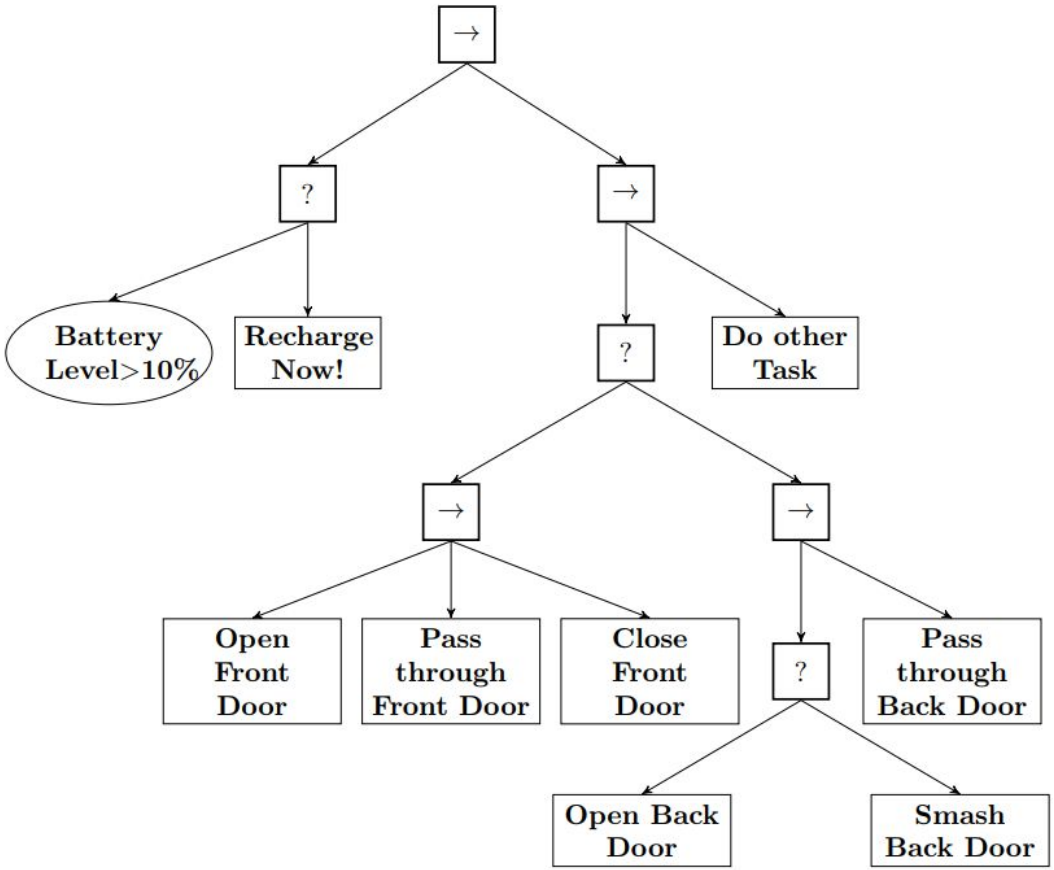
Fig. 1.5: Graphical representation of Action (a), Condition (b), and Decorator (c) node.

BT Summary

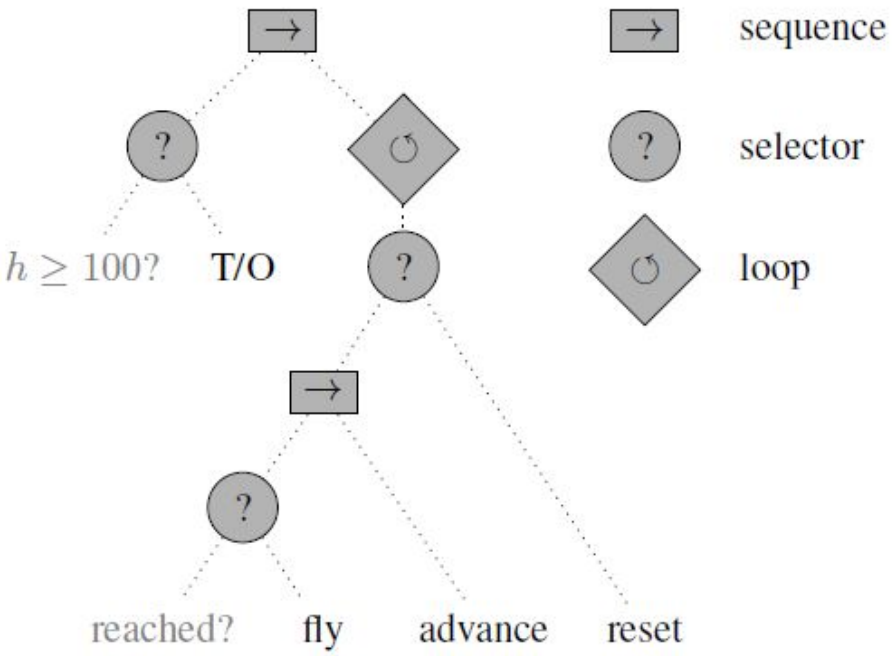
Node type	Symbol	Succeeds	Fails	Running
Fallback	?	If one child succeeds	If all children fail	If one child returns Running
Sequence	$\rightarrow$	If all children succeed	If one child fails	If one child returns Running
Parallel	$\rightrightarrows$	If $\geq M$ children succeed	If $> N - M$ children fail	else
Action	text	Upon completion	If impossible to complete	During completion
Condition	text	If true	If false	Never
Decorator	$\Diamond$	Custom	Custom	Custom

Table 1.1: The node types of a BT.

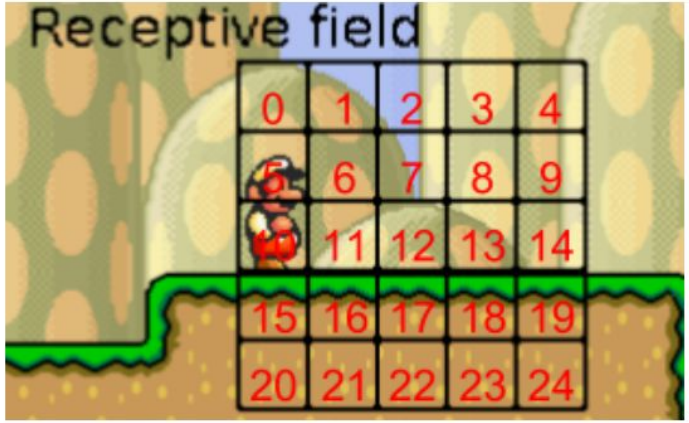
Robotics

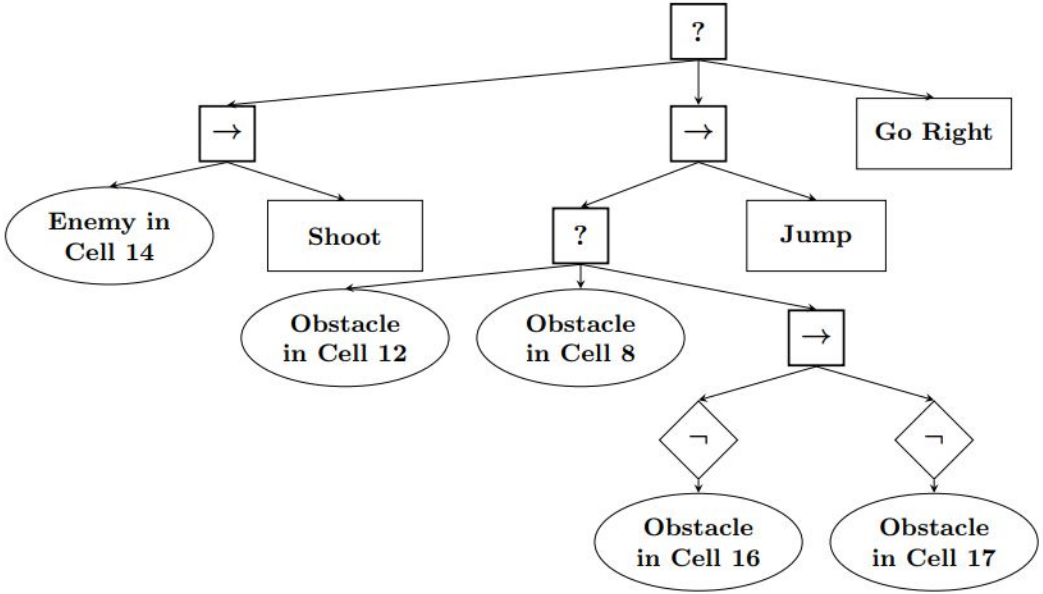

→
?
→
Battery Level>10%
Recharge Now!
?
Do other Task
→
→
Open Front Door
Pass through Front Door
Close Front Door
?
Pass through Back Door
Open Back Door
Smash Back Door

Mission Management

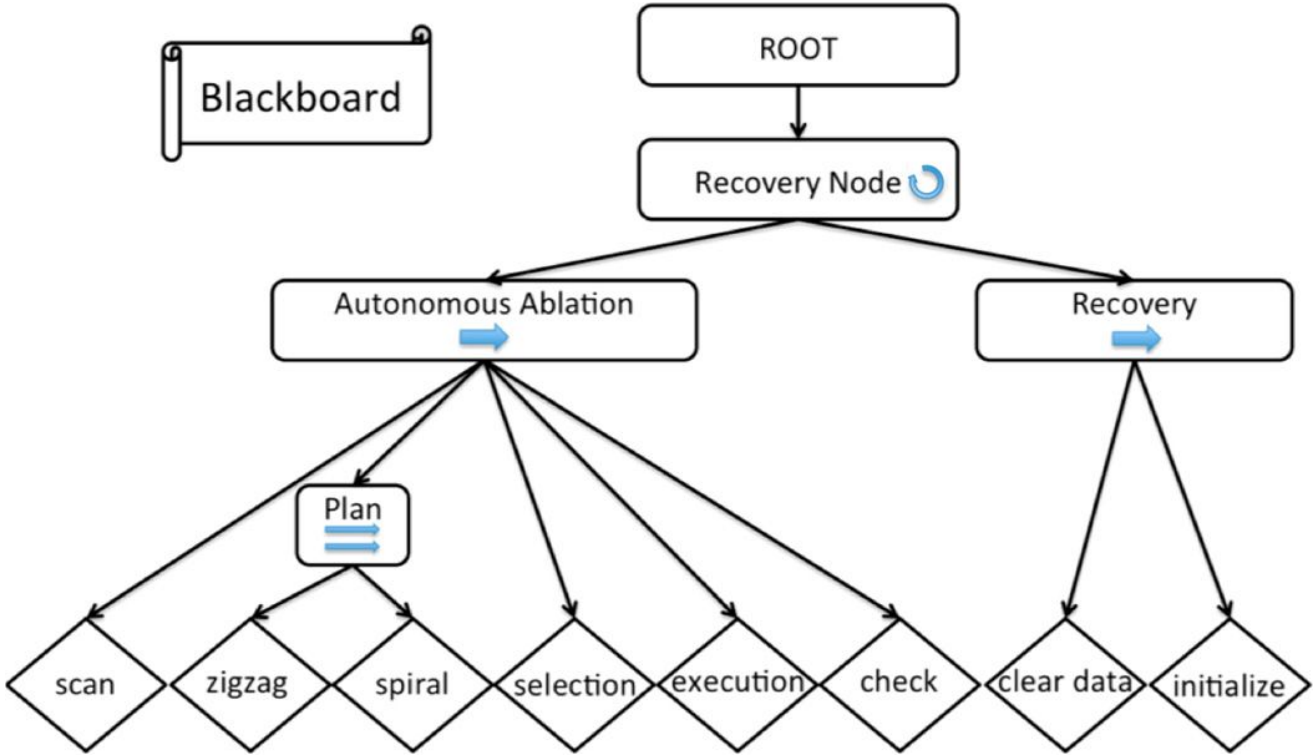


Game

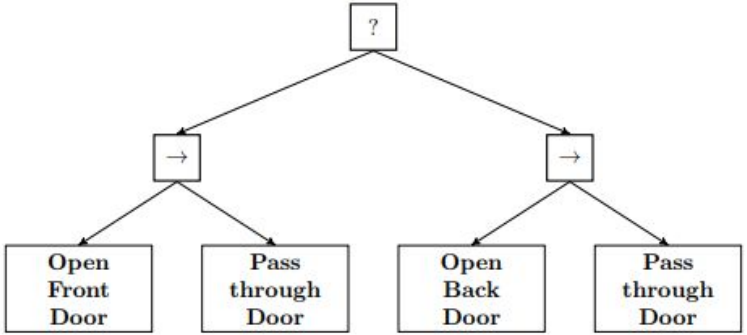

Receptive field
0 1 2 3 4
5 6 7 8 9
10 11 12 13 14
15 16 17 18 19
20 21 22 23 24

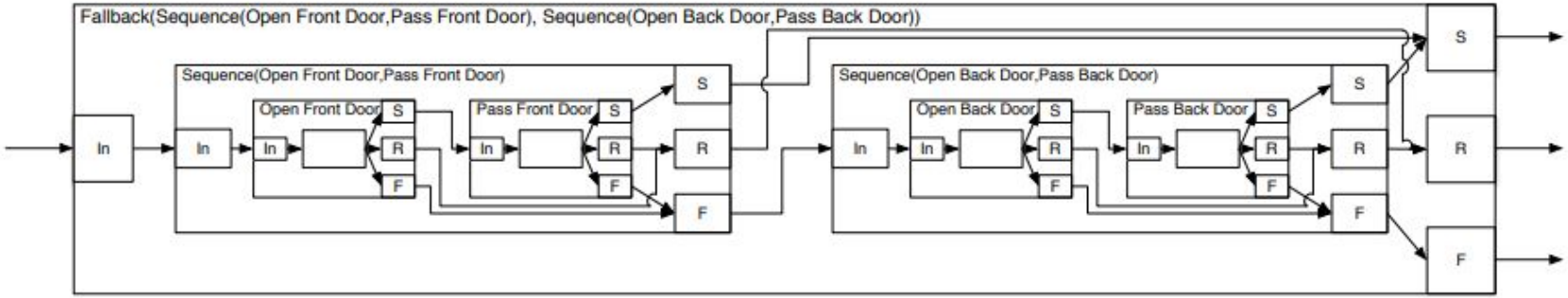

?
→
→
Go Right
Enemy in Cell 14
Shoot
?
Jump
Obstacle in Cell 12
Obstacle in Cell 8
→
¬
¬
Obstacle in Cell 16
Obstacle in Cell 17

Medical

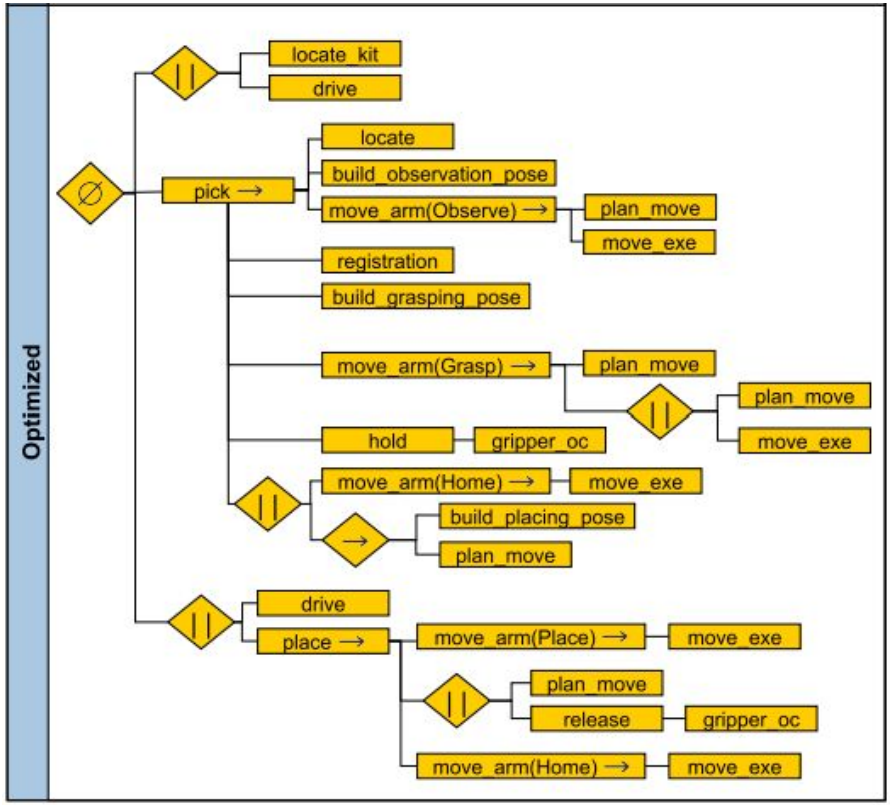

Blackboard
ROOT
Recovery Node
Autonomous Ablation
Recovery
Plan
scan
zigzag
spiral
selection
execution
check
clear data
initialize

BT vs FSM

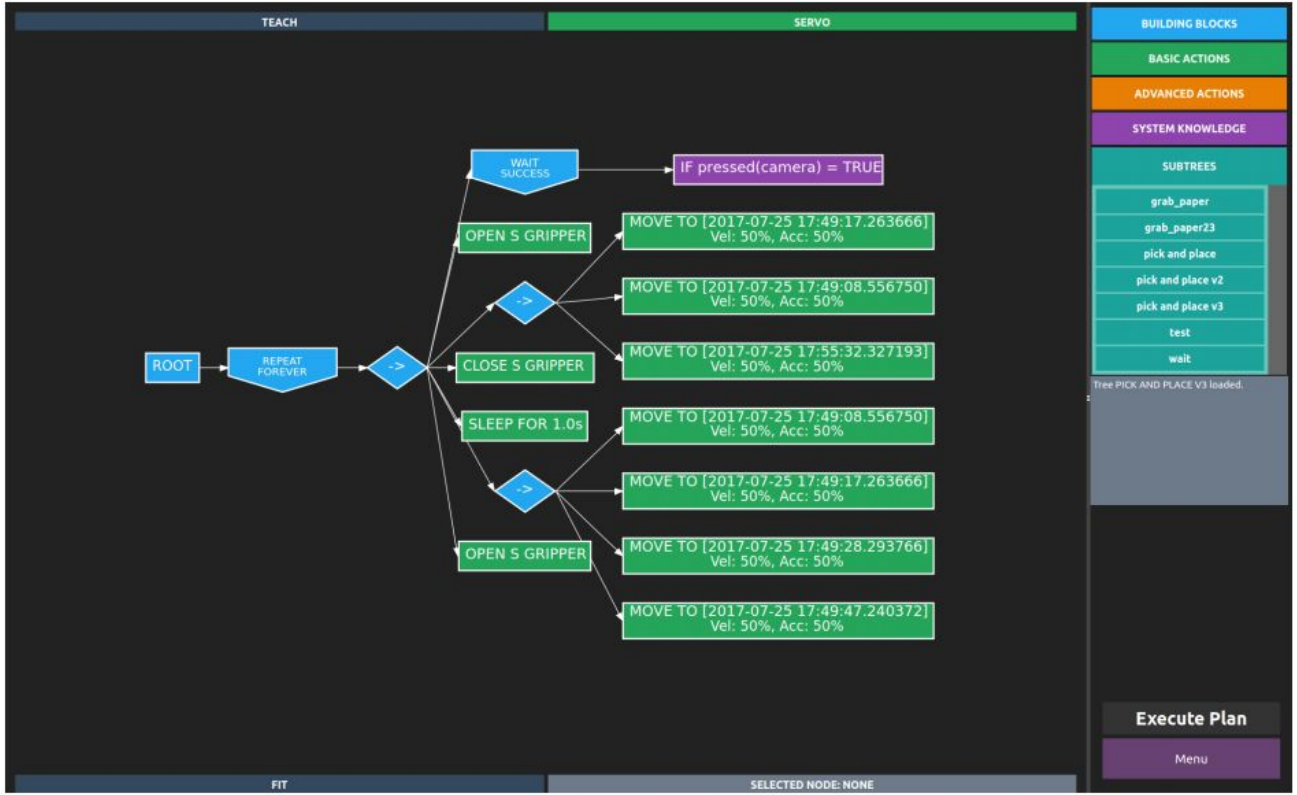

?
→
→
Open Front Door
Pass through Door
Open Back Door
Pass through Door


Fallback(Sequence(Open Front Door,Pass Front Door), Sequence(Open Back Door,Pass Back Door))
Sequence(Open Front Door,Pass Front Door)
Open Front Door
Pass Front Door
Sequence(Open Back Door,Pass Back Door)
Open Back Door
Pass Back Door
In
S
R
F

BT for mission management?



BT for mission management?



BT for mission management?

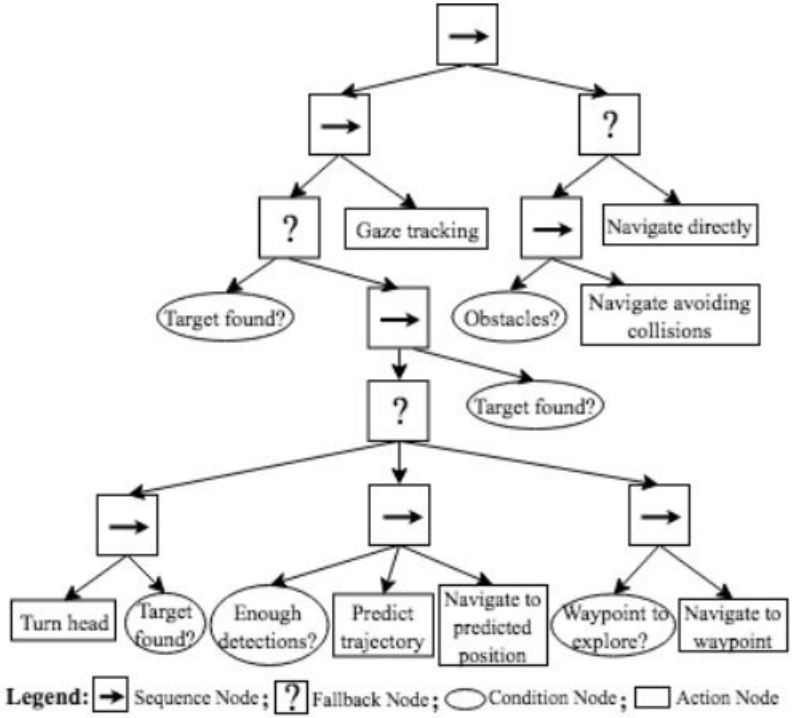
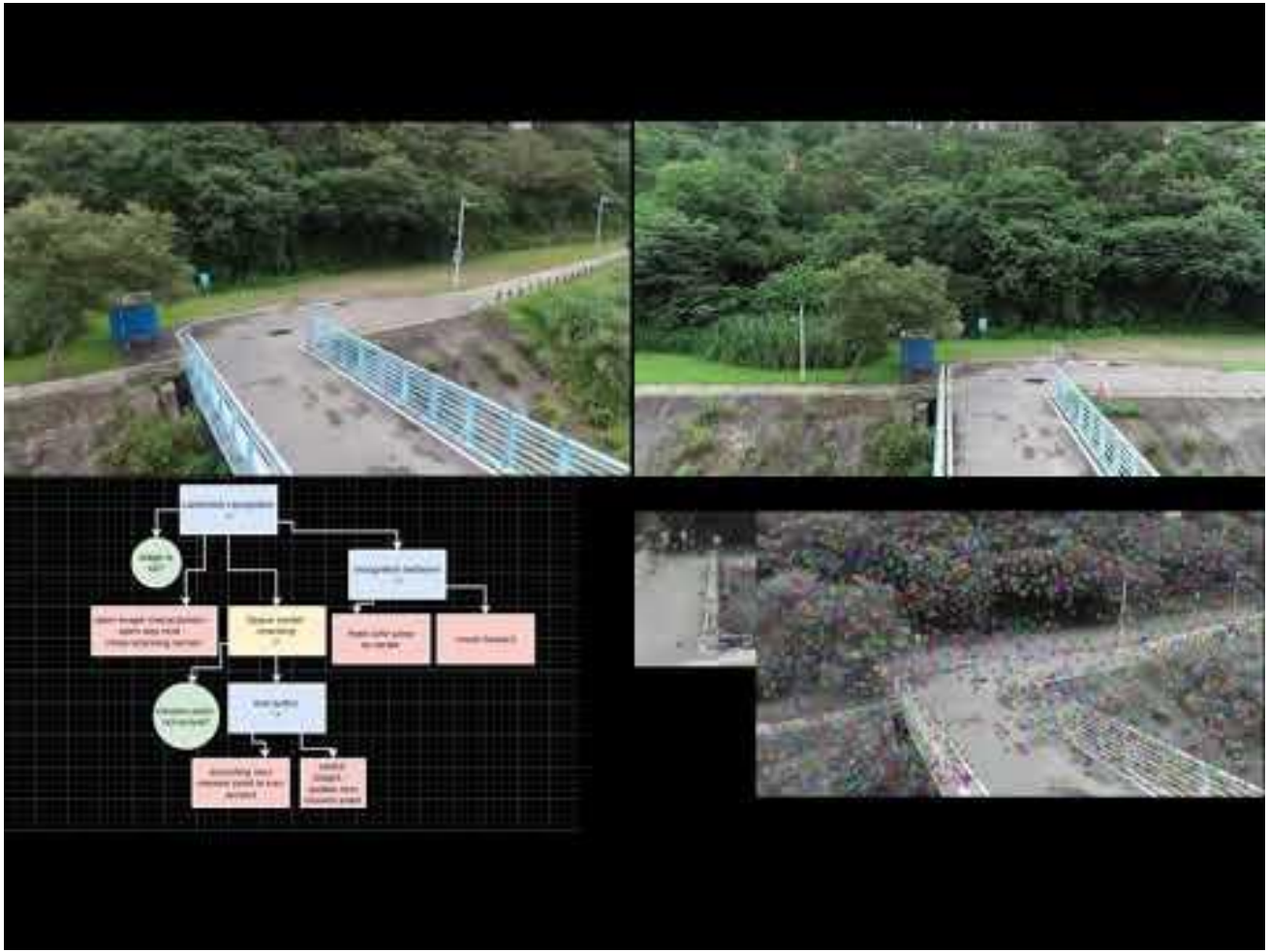


Fig. 2. Behavior-Tree architecture for person-following.



Implementation

```
from behave import condition, action, FAILURE

tree = (
    is_greater_than_10 >> wow_large_number
    | is_between_0_and_10 >> count_from_1
    | failer * repeat(3) * doomed
)
bb = tree.blackboard(10)
while bb.tick() == RUNNING:
    pass

@condition
def is_greater_than_10(x):
    return x > 10

@action
def wow_large_number(x):
    print "WOW, %d is a large number!" % x
```

https://github.com/fuchen/behave

example

See moodle

```
condition: BlueNotFinish
condition: isNotCenter
action: FixedPose
state = Success
```

```
condition: BlueNotFinish
condition: isNotCenter
action: PassAndSwitch
action: FixedPose
state = Success
```