# Attentional Trees And Robotics: Towards an Optimal Executive Framework Meeting High-level Decision Making and Control Layer Features.

1st Pilar de la Cruz
*Intelligent And Interactive Systems Department*
*University of Innsbruck*
Innsbruck, Austria
pilar.de-la-cruz@uibk.ac.at

2nd Matteo Saveriano
*Intelligent And Interactive Systems Department*
*University of Innsbruck*
Innsbruck, Austria
matteo.saveriano@uibk.ac.at

***Abstract*—(Brainstorm) Motivation: HRI. Attentional robot - flexible, if possible event-based. Introduce behavior tree trend. Potential in robotics, yet too high level. Merge. JSON. What is this system providing as new? Why is it convincing?**

***Index Terms*—behavior trees, robotics, event-based, json**

## I. Introduction

### A. Behavior Trees (Brainstorm)

- Ticking - the ability to tick allows for work between executions without multi-threading
- Priority Handling - switching mechansims that allow higher priority interruptions is very natural
- Simplicity - very few core components, making it easy for designers to work with it
- Dynamic - change the graph on the fly, between ticks or from parent behaviours themselves

### B. Behavior Trees Limitations (Brainstorm)

### C. How Limitations can be overcome: JSON (Brainstorm)

JSON is a standard format for data exchange inspired by JavaScript. Generally, JSON is in string or text format. This schema provides (1) simplicity due to its compact description of information, (2) flexibility due to its abstract base representation of tasks and (3) modularity due to the definition of subtrees which can be added and/orremoved on the fly in whatever position of Behavior Tree.

### D. What is this document about (Brainstorm)

### E. Describe Sections Structure and Content

## II. Related Works

Add Matteo's system. Add SoA Behavior Trees.

## III. System Overview

Attentional Trees framework aims at integrating the following features:

1) JSON schema conversion into Behavior Trees
2) Task planner based on Behavior Trees
3) Priority handling of tasks based on the emphasis value [2]
4) Event-based approach of Behavior Tree execution

*JSON Parsing:* A typical JSON schema consists of a list of string objects. The first object element defines the root of the Behavior Tree. The remaining object elements are defined as subtrees and will be added recursively as children from the root. Each object contains properties `name`, `type`, `father`, `children` and `parameters`, which provide with information about the Behavior Tree node types as well as additional attributes for pre-and post-conditioning handling, priority execution and any additional information.

```
{
    "name":"selector01",
    "type":"selector",
    "father": "sequence01",
    "children": ["isReached","reach"],
    "parameters": [["emphasis", 1]]
}
```

### A. Task Planning

### B. Priority Handling

### C. Event-Based Behavior Tree

## Implementation

The framework platform of Attentional Trees is based on the open-source py_trees [4] library. py_trees is compatible with ROS and provides the easiness and quickness of an implementation based in python language. Moreover, decoding operations becomes specially intuitive thanks to the in-built mapping between container and value (JSON objects to Python objects and viceversa) based on Python standard libraries.

## Simulations

In order to test the requirements described in section above, the following demo programs have been implemented[1]

- `json-tree.py`. This program focuses on (1) loading JSON schema and (2) mapping into a Behavior Tree based on PA-BT approach [3].

[1]These are available on:

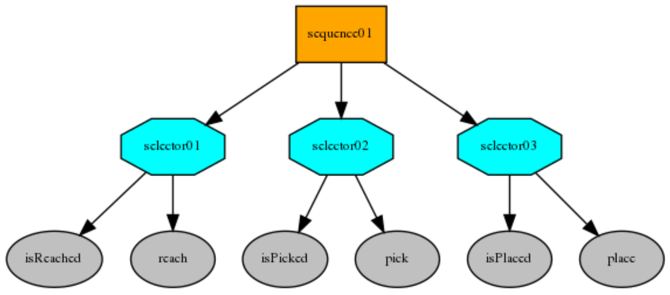

Fig. 1. Tree expansion rendered from json schema rendered in .dot graph

- `emphasis-tree.py`. This program focuses on high-priority handling based on access to emphasis value [1].
- `attentional-tree.py`. This program focuses on event-based task execution of Behavior Trees. Tick signals are not periodically sent, but triggered by special events handled in the control layer.

## Discussion

## References

[1] Add Matteo's paper and page of emphasis description
[2] https://www.guru99.com/python-json.html
[3] Behavior Tree book
[4] https://py-trees.readthedocs.io/en/devel/