Modular Design and Coordination of Computer-Controlled Characters in Games

•••

Thesis Defense – Computer Science Department, University of Antwerp

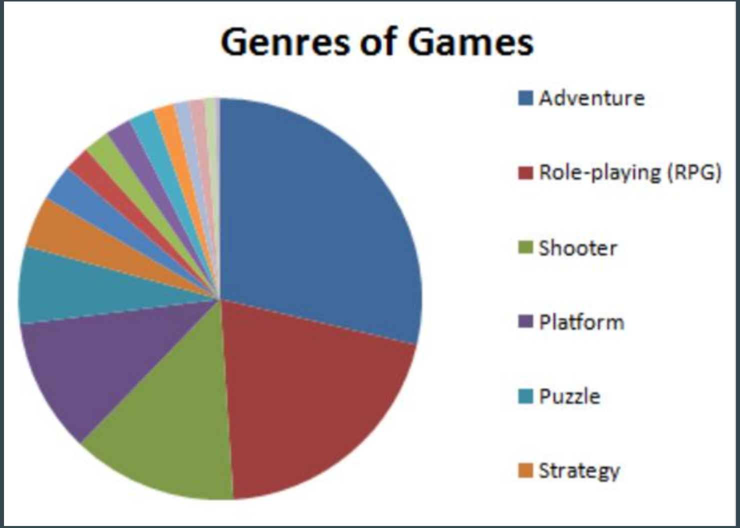
Khemin Van Gestelen

June 26th, 2025

Promotor: Prof. Hans Vangheluwe – Co-promotor: Prof. Clark Verbrugge – Advisor: Joeri Exelmans

Motivation

- The need for believable NPCs
- Popular game genres
- Team-based game
- Autonomy and Coordination

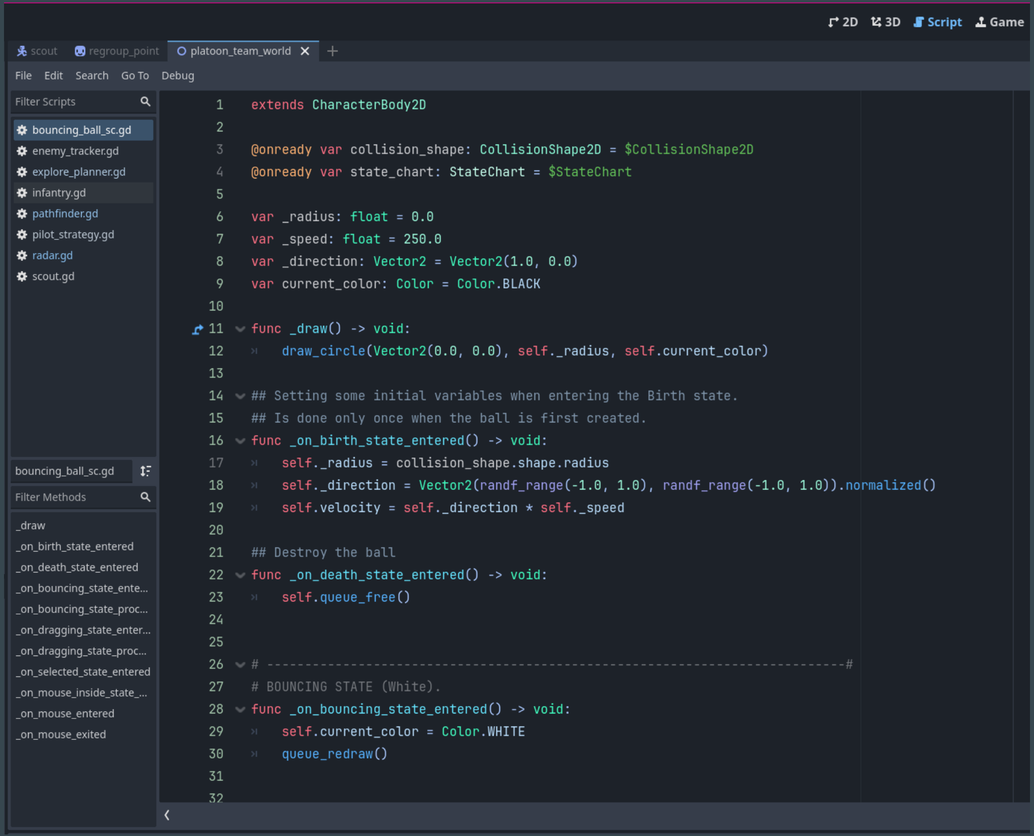


[1]

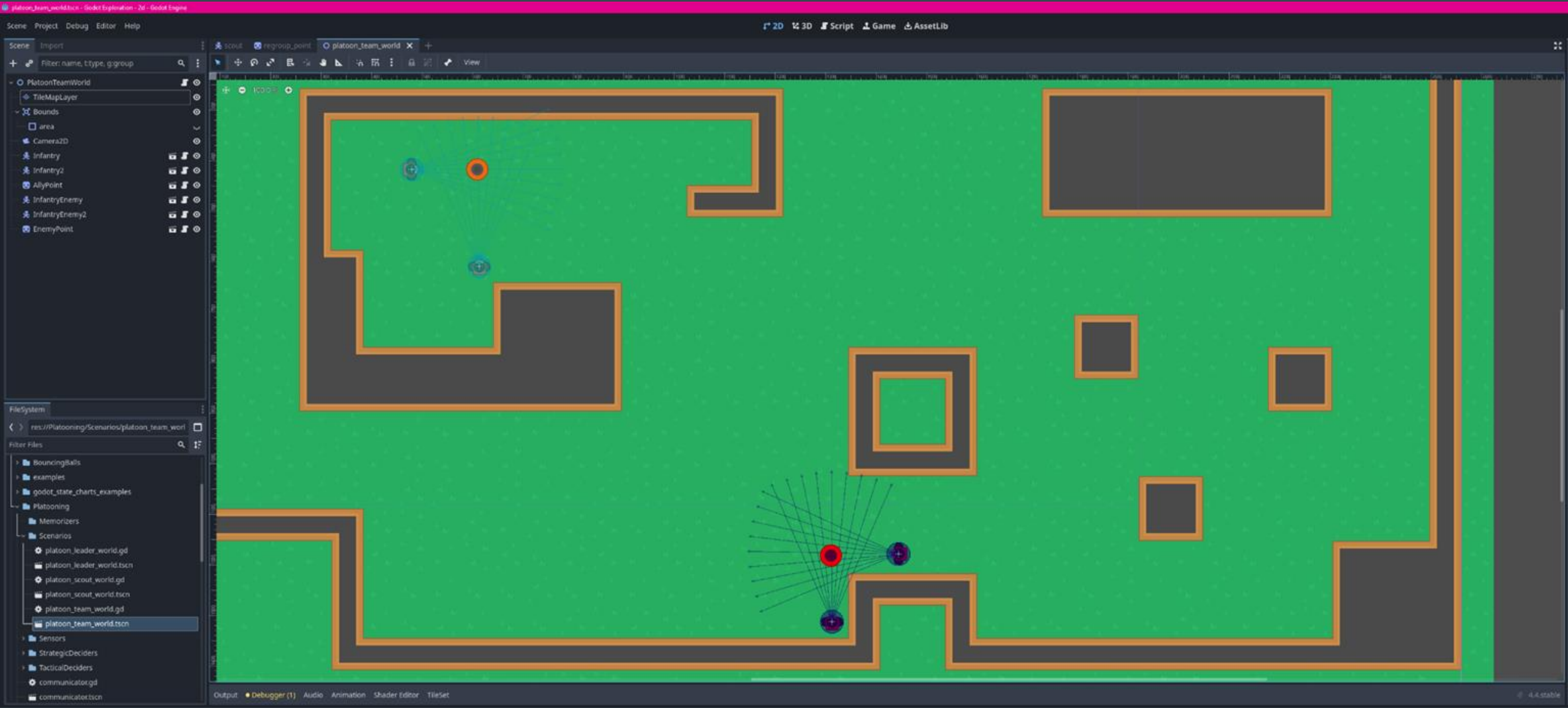
[1] A. A. Qaffas, “An operational study of video games’ genres,” International Journal of Interactive Mobile Technologies (iJIM), vol. 14, no. 15, pp. 175–194, 2020.

The Godot engine

- Nodes
- Scenes
- SceneTree
- Behavior through code
  - GDScript

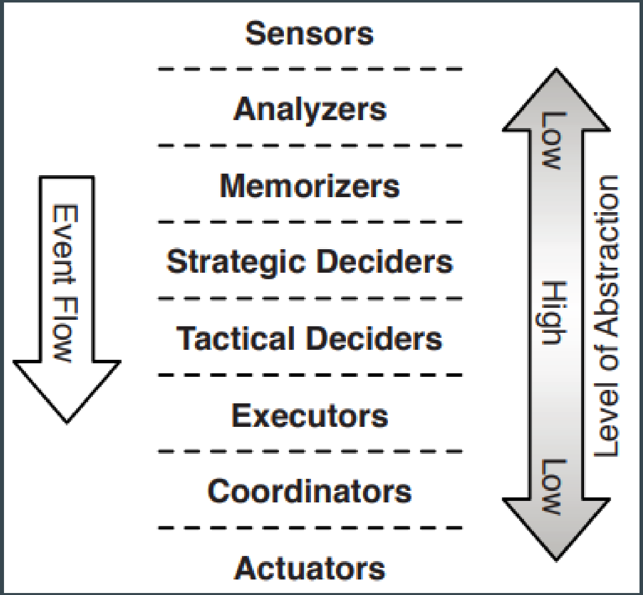


```
extends CharacterBody2D

@onready var collision_shape: CollisionShape2D = $CollisionShape2D
@onready var state_chart: StateChart = $StateChart

var _radius: float = 0.0
var _speed: float = 250.0
var _direction: Vector2 = Vector2(1.0, 0.0)
var current_color: Color = Color.BLACK

func _draw() -> void:
	draw_circle(Vector2(0.0, 0.0), self._radius, self.current_color)

## Setting some initial variables when entering the Birth state.
## Is done only once when the ball is first created.
func _on_birth_state_entered() -> void:
	self._radius = collision_shape.shape.radius
	self._direction = Vector2(randf_range(-1.0, 1.0), randf_range(-1.0, 1.0)).normalized()
	self.velocity = self._direction * self._speed

## Destroy the ball
func _on_death_state_entered() -> void:
	self.queue_free()


# ----------------------------------------------------------------------------#
# BOUNCING STATE (White).
func _on_bouncing_state_entered() -> void:
	self.current_color = Color.WHITE
	queue_redraw()

```


platoon_team_world.tscn - Godot Exploration - 2d - Godot Engine
Scene Project Debug Editor Help
2D 3D Script Game AssetLib
Scene Import
scout regroup_point platoon_team_world
View
PlatoonTeamWorld
TileMapLayer
Bounds
area
Camera2D
Infantry
Infantry2
AllyPoint
InfantryEnemy
InfantryEnemy2
EnemyPoint
FileSystem
res://Platooning/Scenarios/platoon_team_worl
BouncingBalls
examples
godot_state_charts_examples
Platooning
Memorizers
Scenarios
platoon_leader_world.gd
platoon_leader_world.tscn
platoon_scout_world.gd
platoon_scout_world.tscn
platoon_team_world.gd
platoon_team_world.tscn
Sensors
StrategicDeciders
TacticalDeciders
communicator.gd
communicator.tscn
Output Debugger (1) Audio Animation Shader Editor TileSet
4.4.stable

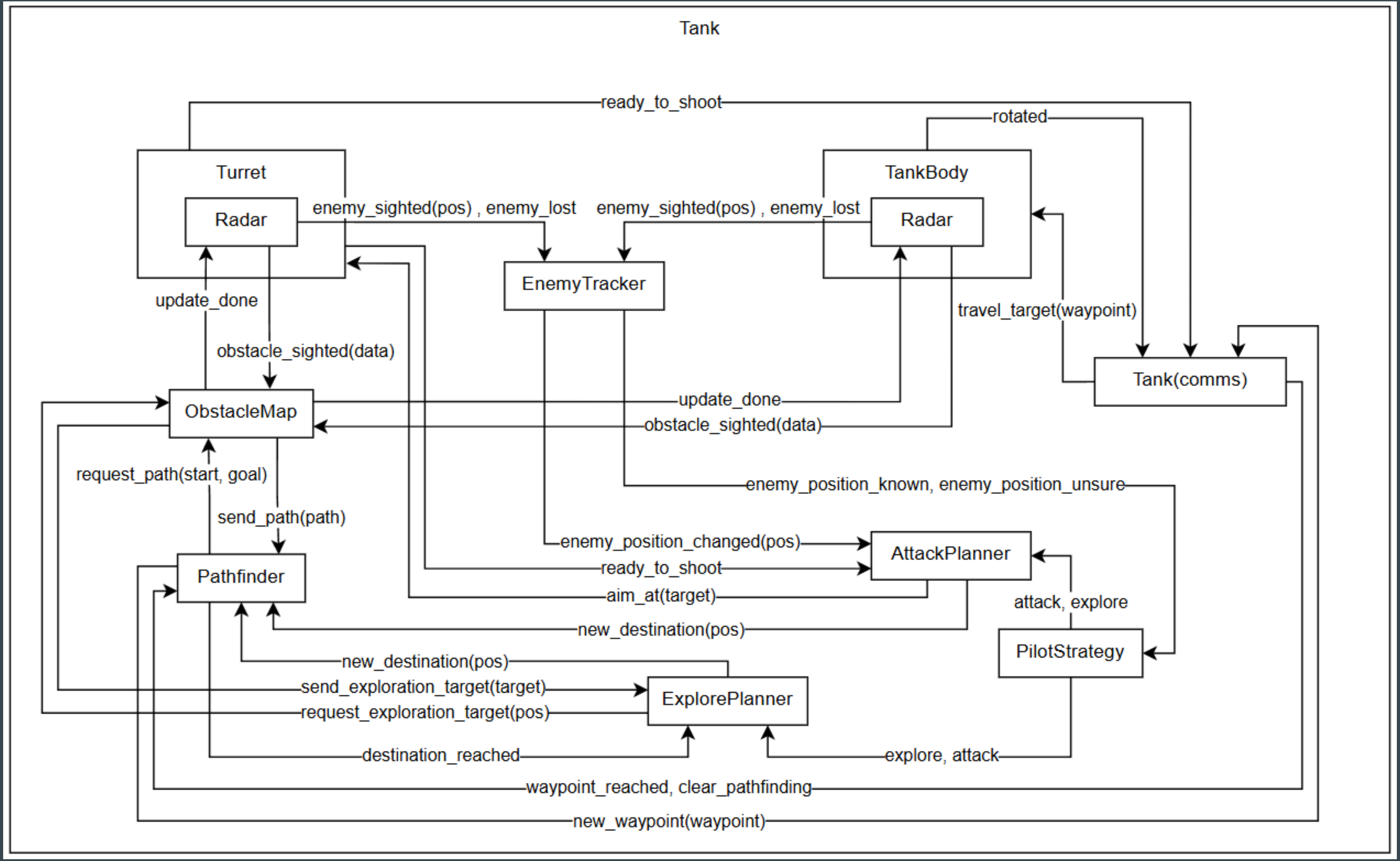
Modular Design

- Understandability
- Reusability
- Godot's modularity
- Appropriate formalism
- Inspiration for approach
  - Model-based design of computer-controlled game character behavior [2]
  - Reusable components for artificial intelligence in computer games [3]



[2]

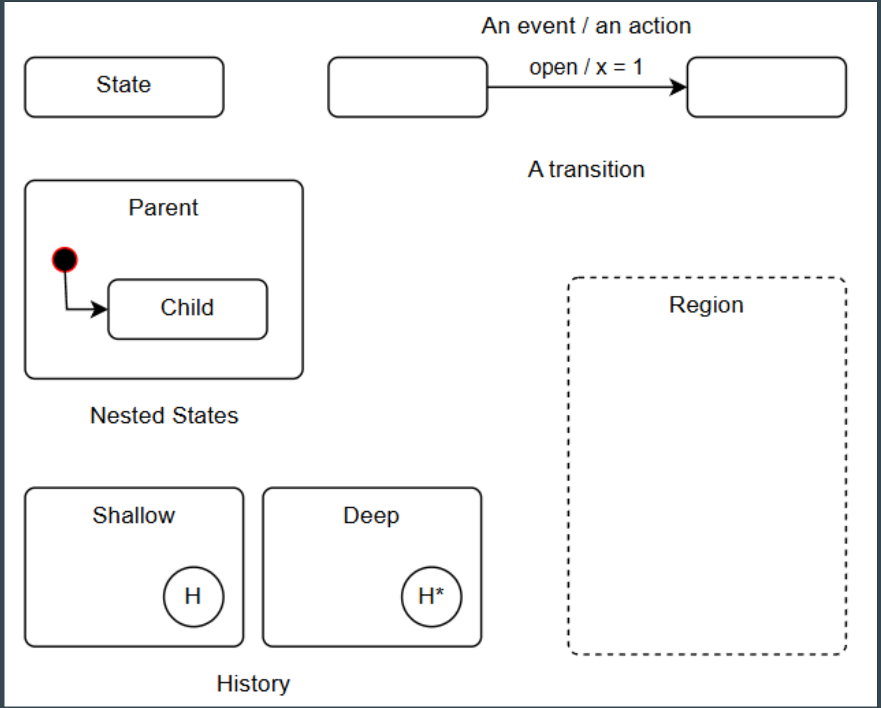
[2] J. Kienzle, A. Denault, and H. Vangheluwe, "Model-based design of computer-controlled game character behavior," MoDELS 2007, LNCS 4735, pp. 650—665, 2007.
[3] C. Dragert, J. Kienzle, and C. Verbrugge, "Reusable components for artificial intelligence in computer games," GAS, pp. 35–41, 2012.



Tank
ready_to_shoot
rotated
Turret
Radar
enemy_sighted(pos) , enemy_lost
enemy_sighted(pos) , enemy_lost
TankBody
Radar
EnemyTracker
update_done
obstacle_sighted(data)
travel_target(waypoint)
Tank(comms)
ObstacleMap
update_done
obstacle_sighted(data)
request_path(start, goal)
send_path(path)
enemy_position_known, enemy_position_unsure
enemy_position_changed(pos)
AttackPlanner
ready_to_shoot
Pathfinder
aim_at(target)
attack, explore
new_destination(pos)
PilotStrategy
new_destination(pos)
send_exploration_target(target)
ExplorePlanner
request_exploration_target(pos)
destination_reached
explore, attack
waypoint_reached, clear_pathfinding
new_waypoint(waypoint)

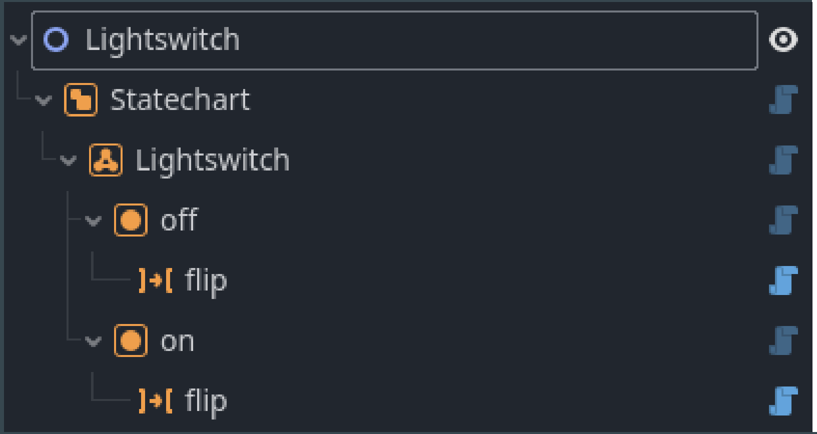
Statecharts

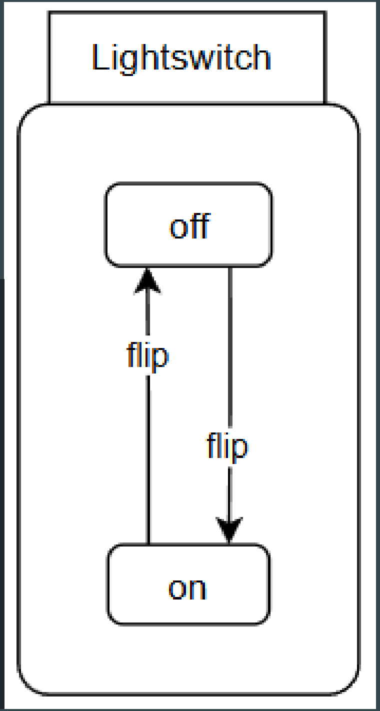
- Finite State Machines
- Hierarchy
- Concurrency
- Timed behavior
- History



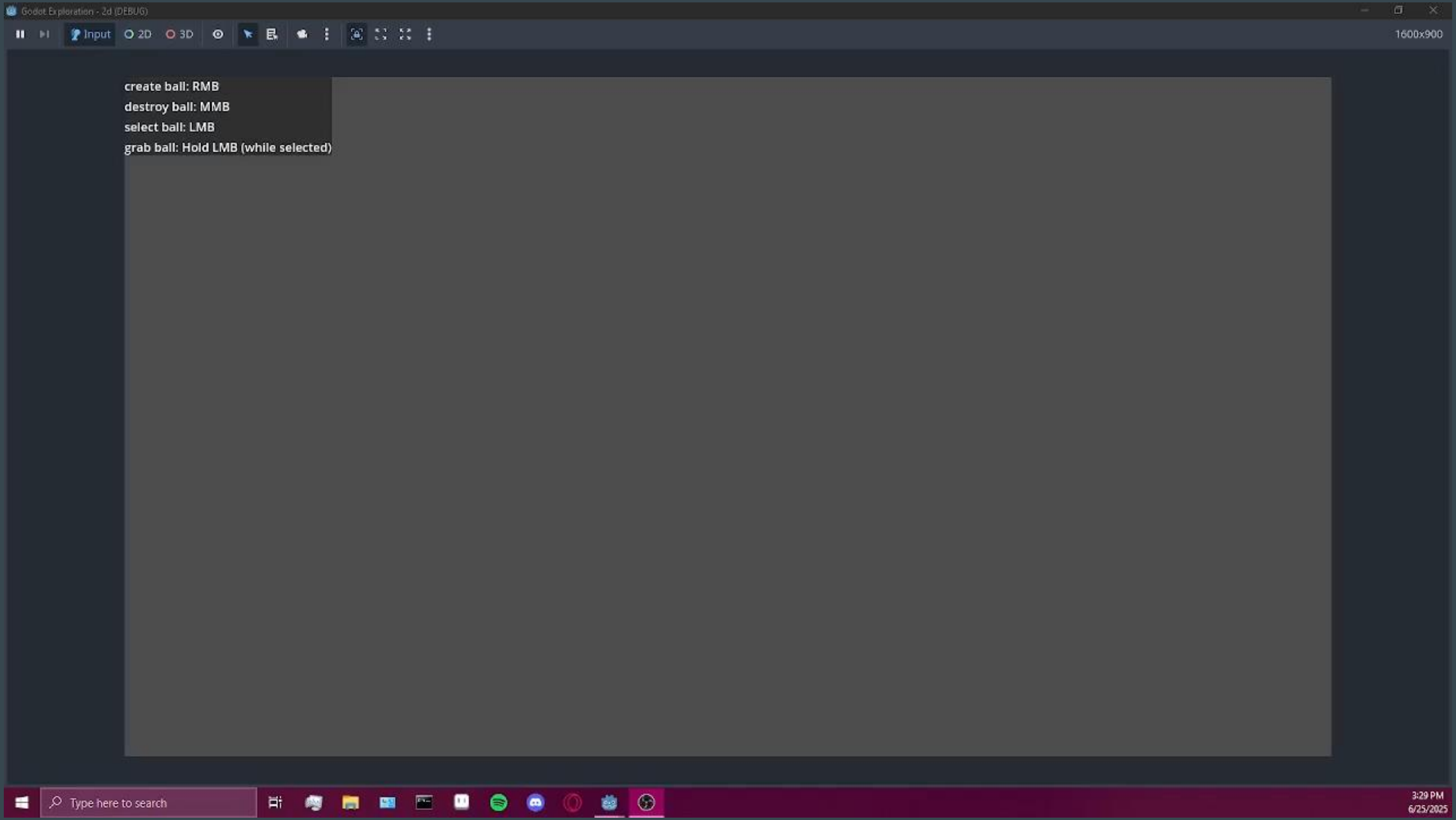
Statecharts in Godot

- Godot Statechart Extension
  - developed by Godot community developer





Bouncing Ball Demo

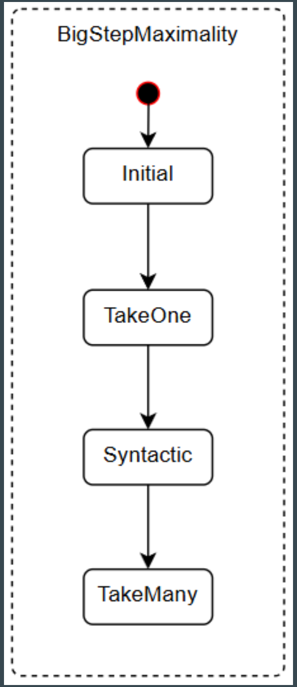


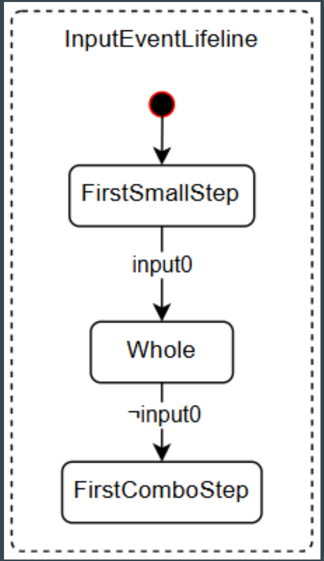
create ball: RMB
destroy ball: MMB
select ball: LMB
grab ball: Hold LMB (while selected)

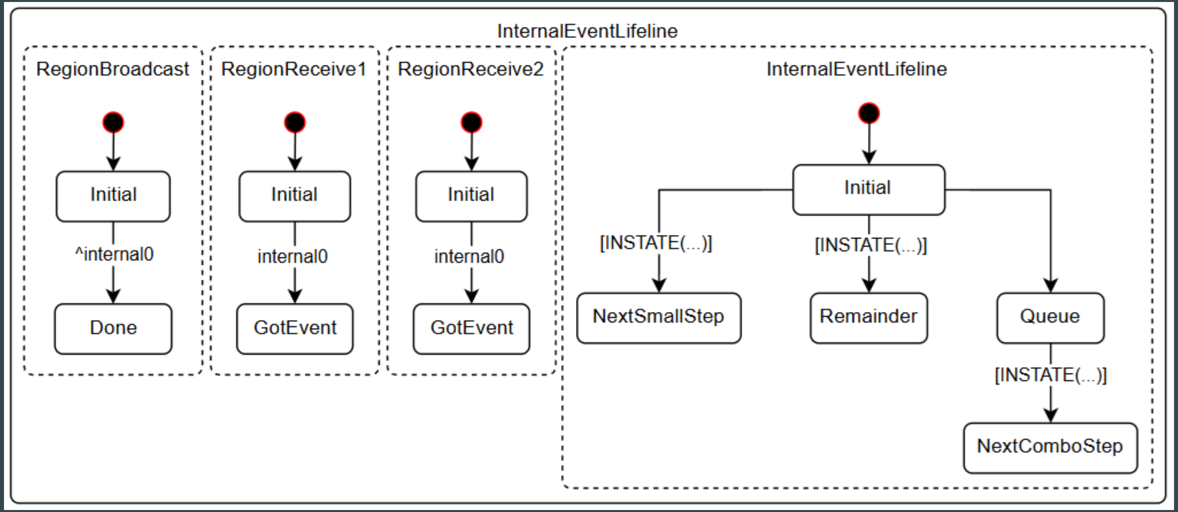
Semantic analysis

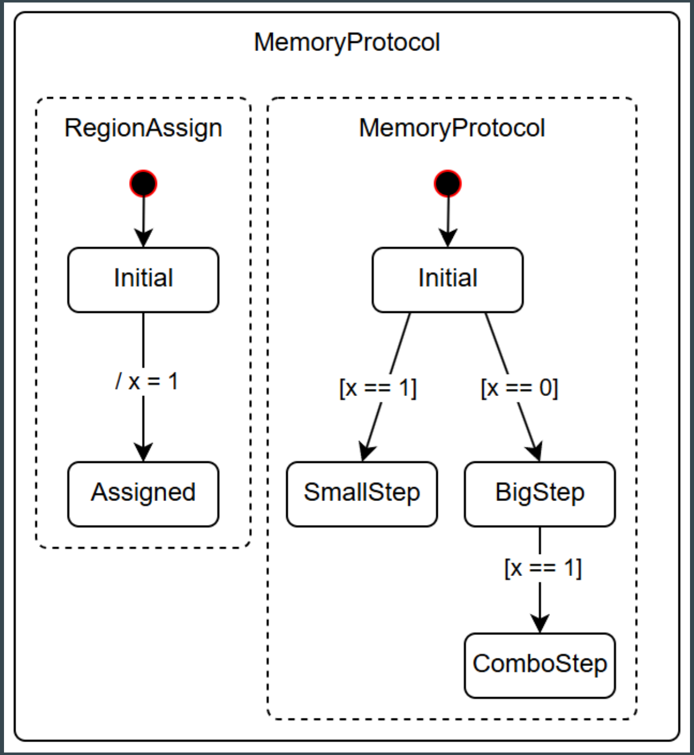
- Limited documentation
- Testing statechart [4]
    - Orthogonal regions testing different semantics
- Additional small tests

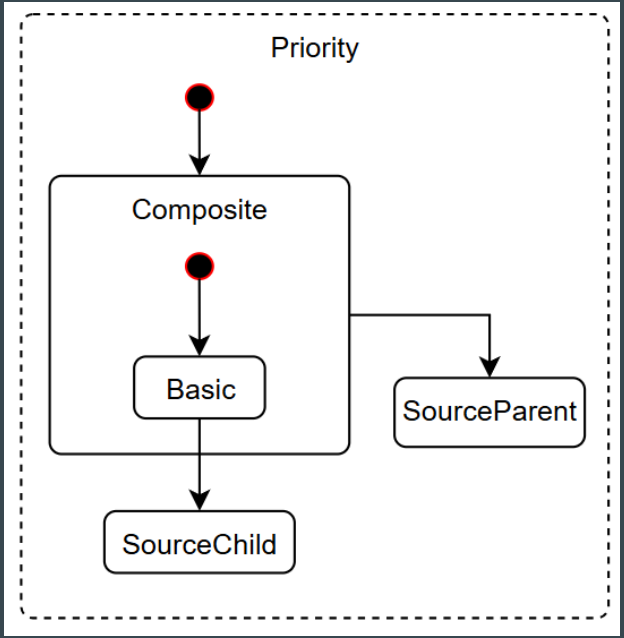
[4] J. Exelmans, S. Van Mierlo, and H. Vangheluwe, “A statecharts interpreter and compiler with semantic variability,” MODELS ’22: Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, pp. 722–727, 2022.

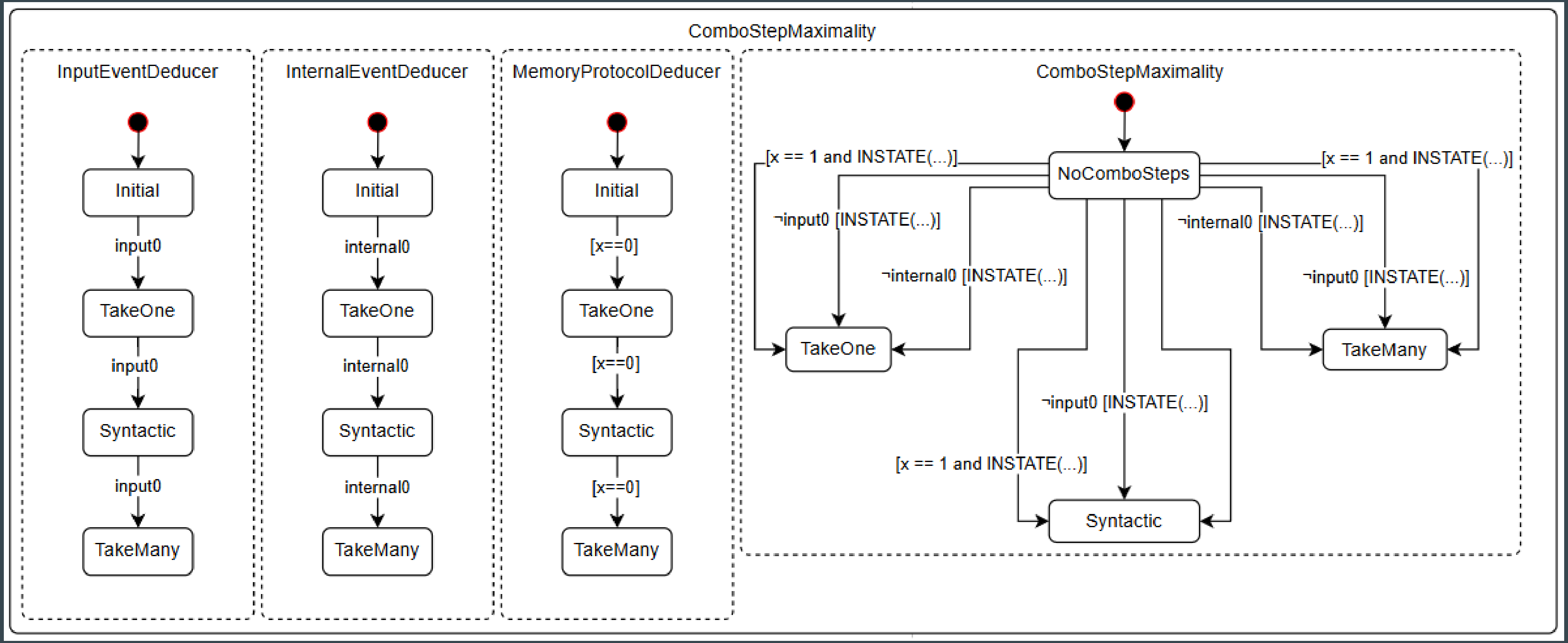

BigStepMaximality
Initial
TakeOne
Syntactic
TakeMany


InputEventLifeline
FirstSmallStep
input0
Whole
¬input0
FirstComboStep

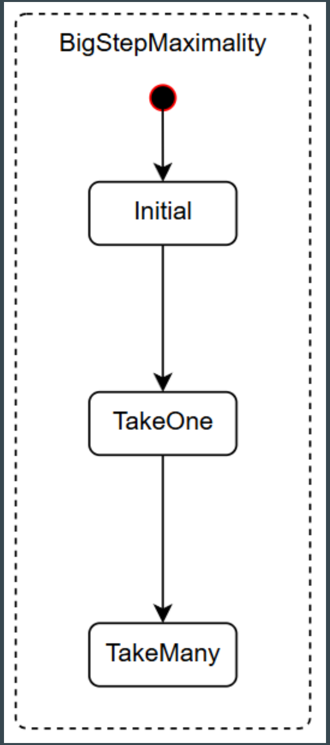

InternalEventLifeline
RegionBroadcast
Initial
^internal0
Done
RegionReceive1
Initial
internal0
GotEvent
RegionReceive2
Initial
internal0
GotEvent
InternalEventLifeline
Initial
[INSTATE(...)]
NextSmallStep
[INSTATE(...)]
Remainder
Queue
[INSTATE(...)]
NextComboStep

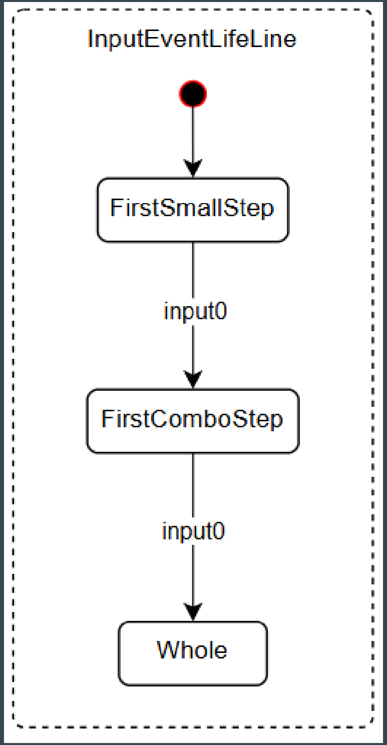

MemoryProtocol
RegionAssign
Initial
/ x = 1
Assigned
MemoryProtocol
Initial
[x == 1]
[x == 0]
SmallStep
BigStep
[x == 1]
ComboStep


Priority
Composite
Basic
SourceParent
SourceChild


ComboStepMaximality
InputEventDeducer
Initial
input0
TakeOne
input0
Syntactic
input0
TakeMany
InternalEventDeducer
Initial
internal0
TakeOne
internal0
Syntactic
internal0
TakeMany
MemoryProtocolDeducer
Initial
[x==0]
TakeOne
[x==0]
Syntactic
[x==0]
TakeMany
ComboStepMaximality
NoComboSteps
[x == 1 and INSTATE(...)]
[x == 1 and INSTATE(...)]
¬input0 [INSTATE(...)]
¬internal0 [INSTATE(...)]
¬internal0 [INSTATE(...)]
¬input0 [INSTATE(...)]
TakeOne
TakeMany
¬input0 [INSTATE(...)]
[x == 1 and INSTATE(...)]
Syntactic

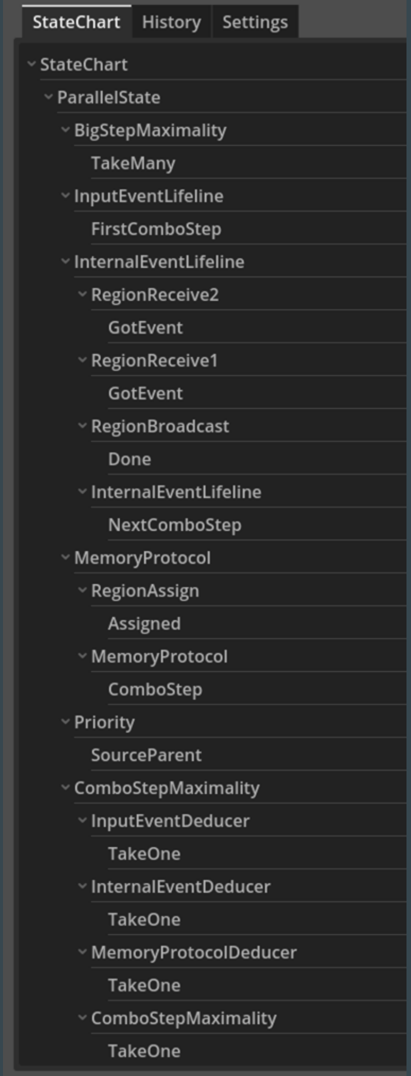
Alterations





Semantic results

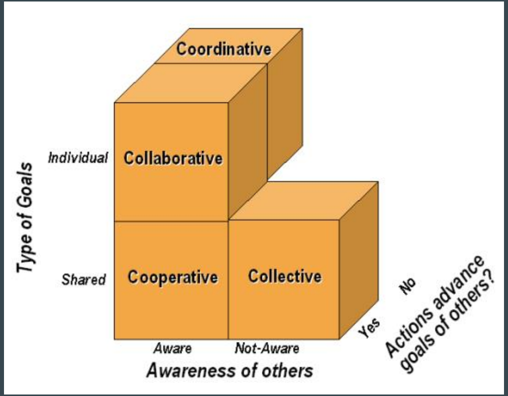
- Transition semantics
- Event lifetime
- Combo-Step



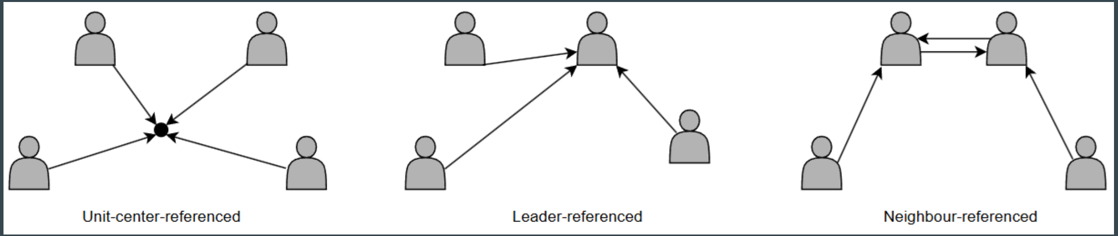
```
StateChart
  ParallelState
    BigStepMaximality
      TakeMany
    InputEventLifeline
      FirstComboStep
    InternalEventLifeline
      RegionReceive2
        GotEvent
      RegionReceive1
        GotEvent
      RegionBroadcast
        Done
      InternalEventLifeline
        NextComboStep
    MemoryProtocol
      RegionAssign
        Assigned
      MemoryProtocol
        ComboStep
    Priority
      SourceParent
    ComboStepMaximality
      InputEventDeducer
        TakeOne
      InternalEventDeducer
        TakeOne
      MemoryProtocolDeducer
        TakeOne
      ComboStepMaximality
        TakeOne
```

Coordination and Communication

- Distributed Intelligence [5]
- Formation Control [6]



[5]

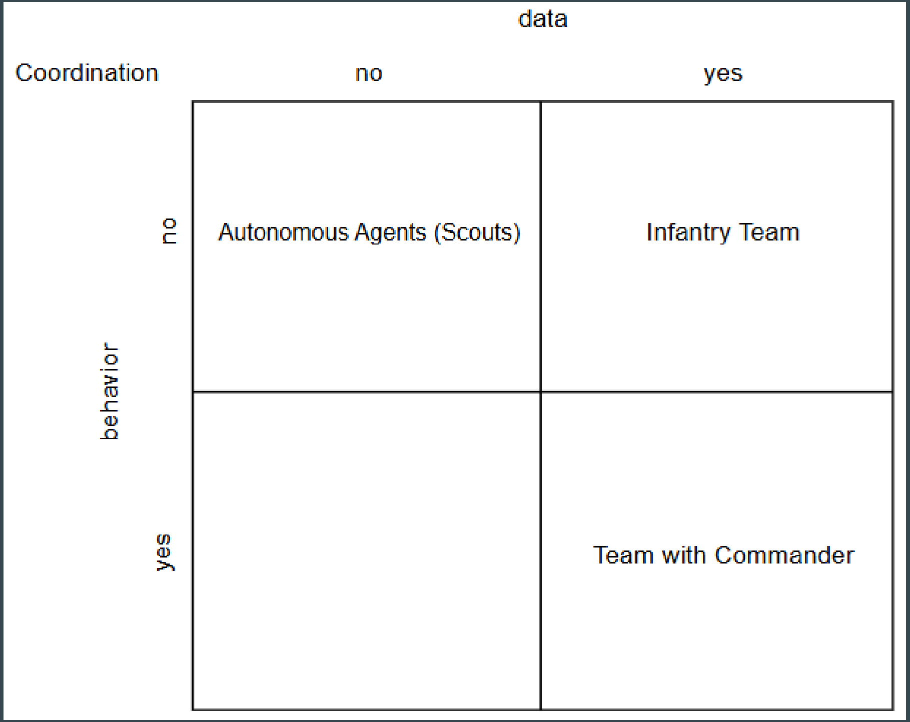


[6]

[5] L. E. Parker, “Distributed intelligence: Overview of the field and its application in multi-robot systems,” Association for the Advancement of Artificial Intelligence, 2007.
[6] T. Balch and R. C. Arkin, “Behavior-based formation control for multirobot teams,” IEEE Transactions on Robotics and Automation, vol. 14, no. 6, pp. 926–939, 1998.

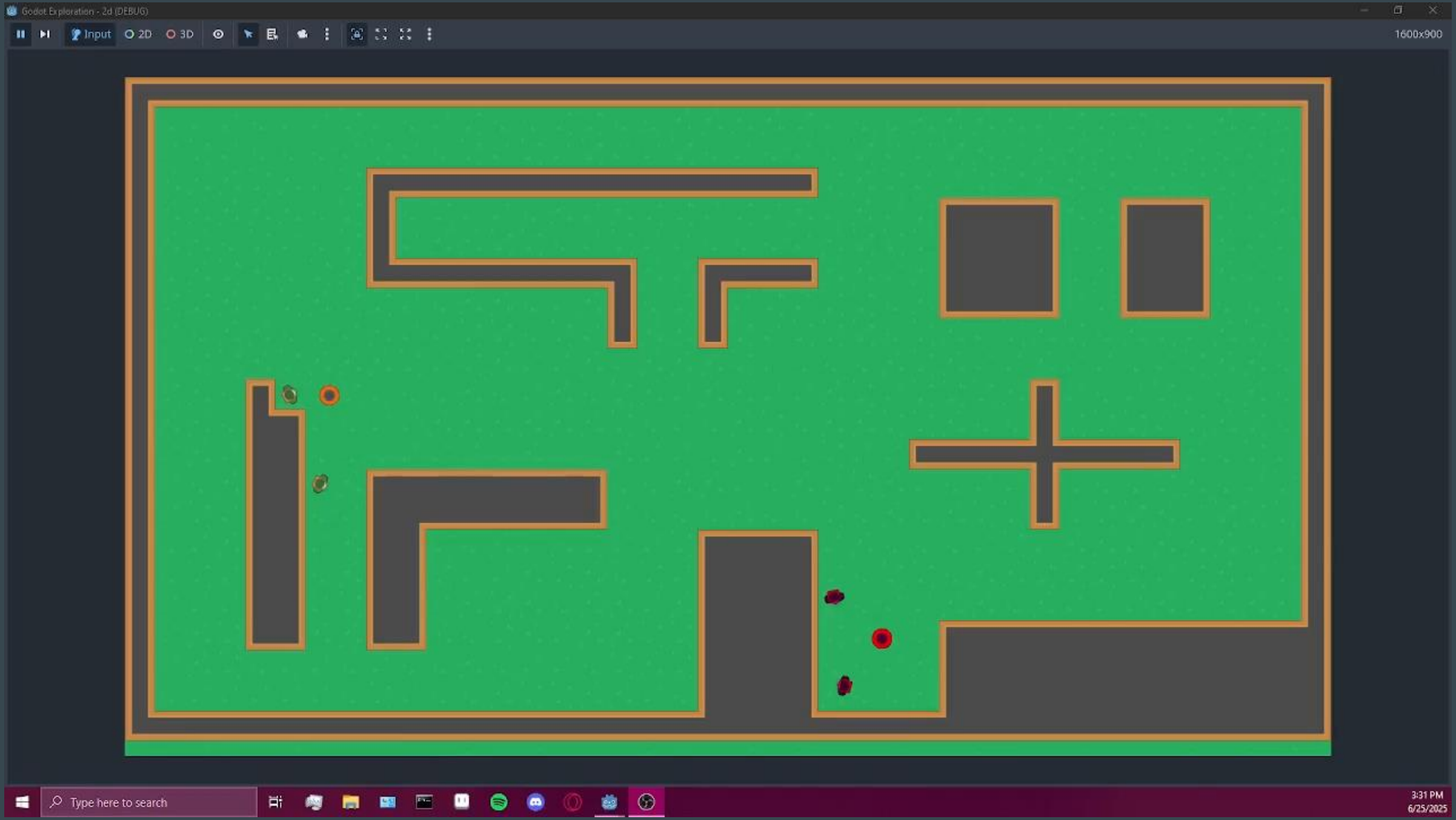
Practical Experiments

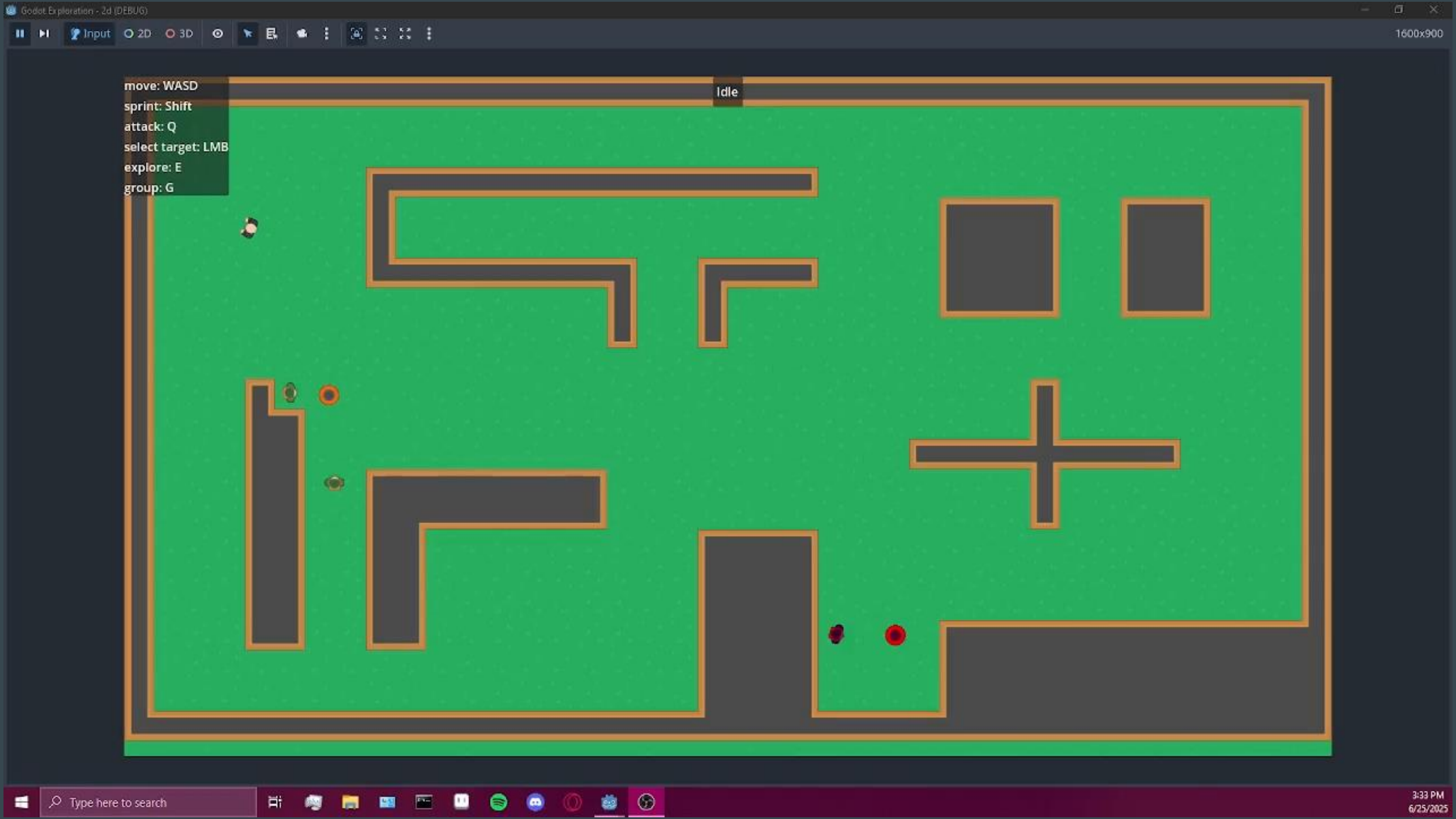
- Interaction
- Coordination and communication



Coordination		data: no	data: yes
behavior	no	Autonomous Agents (Scouts)	Infantry Team
behavior	yes		Team with Commander

Coordination Experiments Demos





move: WASD
sprint: Shift
attack: Q
select target: LMB
explore: E
group: G

Conclusion

- Modular design
- Communicating objects
    - behavior modelled with Statecharts
- Non-player characters in a game

Contributions

- Semantic analysis
- Exploration of modular design in the Godot engine

Future Work

- Evaluation – parameterization
    - playability (include human player)
- Coordination
    - communication, formations, tactics
- Coordination problems
    - friendly fire