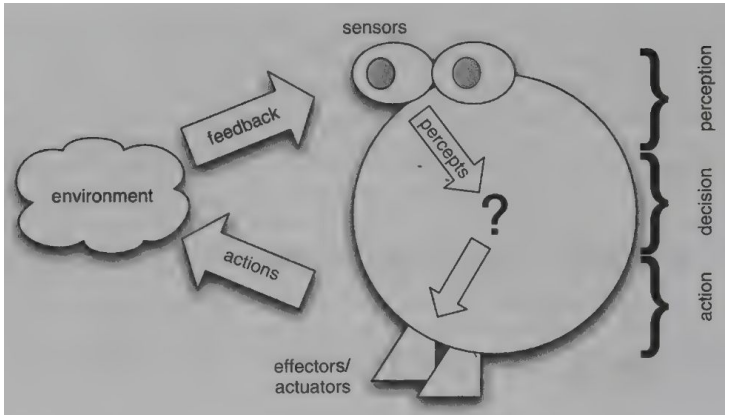
Applications of Autonomous BDI Agents in Robotics

Bardia Parmoun

Carleton University

What are BDI agents?

- "An agent is a **computer system**, situated in some **environment**, that is capable of **flexible autonomous action** in order to meet its **design objectives"**
  - Jennings et al
- Autonomous robots are a great example!
- Popular ways to design robots: imperative programming, state machines, subsumption architecture, etc.
- Proposing **BDI**:
  - **BELIEFS**: agent's knowledge about the world
  - **DESIRES**: the goals for the agent
  - **INTENTIONS**: agent's plans to for achieving its goals



How to write BDI agents?

- **AgentSpeak:** a theoretical formal language for BDI
  - Formal notation for the agent's beliefs, goals, and plans
- **Jason:** one of its most popular implementations!
  - The agent repeatedly perceives, reasons, and acts!

Beliefs

Goals

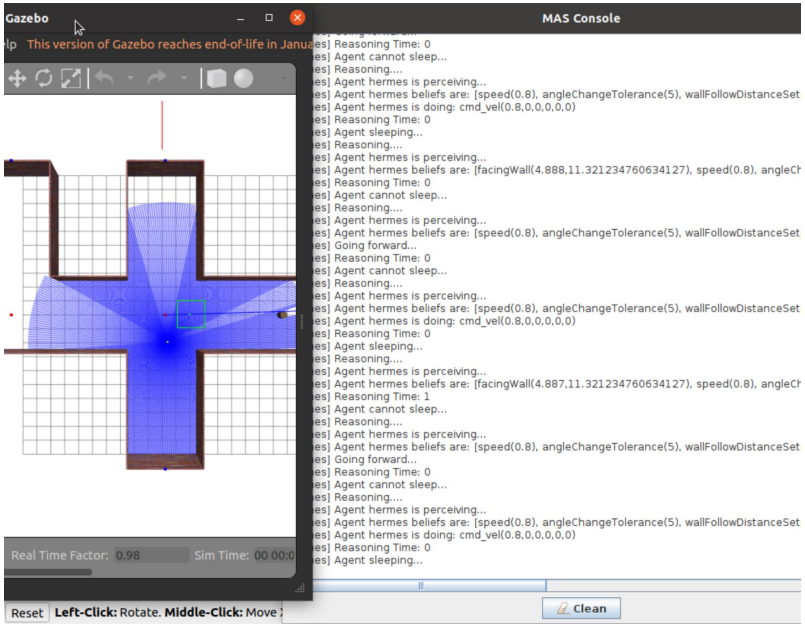
Rules

Plans

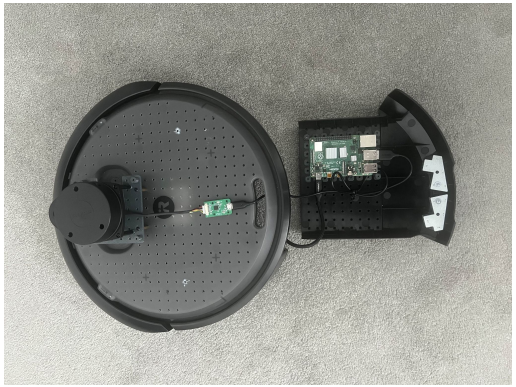
```
wallFollowAimAngle(45).
wallFollowDistanceSetpoint(0.5).
!wallFollow(0.8, 32.0).
tooFarFromWall(WallDistance) :-
    wallFollowDistanceSetpoint(SETPOINT) &
    WallDistance > SETPOINT.
tooCloseToWall(WallDistance) :-
    wallFollowDistanceSetpoint(SETPOINT) &
    WallDistance < SETPOINT.
+!wallFollow(WallDistance, WallAngle):
    not(tooFarFromWall(WallDistance)) &
    not(tooCloseToWall(WallDistance))
    <-
    .print("Following the wall...");
    !turn(WallAngle).
+!wallFollow(WallDistance, WallAngle):
    tooFarFromWall(WallDistance) &
    wallFollowAimAngle(WALL_FOLLOW_AIM_ANGLE)
    <-
    .print("Too far from the wall...");
    !turn(-1 * WALL_FOLLOW_AIM_ANGLE +
          WallAngle).
+!wallFollow(WallDistance, WallAngle):
    tooCloseToWall(WallDistance) &
    wallFollowAimAngle(WALL_FOLLOW_AIM_ANGLE)
    <-
    .print("Too close to the wall...");
    !turn(WALL_FOLLOW_AIM_ANGLE + WallAngle).
```

Jason example!

Hermes: a campus mail delivery robot!



Gazebo Simulator



Physical Robot

Demos!

Carleton University

Findings

- Jason is capable of handling **fast paced real-time** environments.
- **Complex** robotic applications can be implemented in Jason.
- BDI could be used **in conjunction** with well established robotic design patterns such as the subsumption architecture and state machines.
- Jason make the robot implementation **transparent** while still being **efficient**.
- BDI robots are **explainable!**