
# COMP 3004 Designing Intelligent Agents

**Student name:** Grigoriy Kirpa

**Student ID:** 20113717

## Introduction

Due to rapid technological advance in the past several decades, robotic intelligent agents progressively becoming more common in daily life. Although a wide range of autonomous solutions exists for majority of tasks in nearly all domains, each robotic agent has limited physical capabilities. In significant number of cases, a multiagent approach is required to fulfill the needs of the production. For those systems to be viable, some coordination and communication system must be implemented. Although, all control models are categorized into only 2 groups: 1) supervised 2) distributed, choosing between these two approaches is a complicated task. Let's say we have a robotic vacuum cleaner environment that requires a multiagent approach. Given two different systems one being centralized and the other – decentralized, which one is better? In what particular aspects is it better? Does the performance difference scale? And most importantly is it even necessary to have a control system, or is it possible to just use a numerous independent agents and that will suffice? This project aims to answer those questions by implementing both control models and comparing them between each other as well as to some baseline model that doesn't make use of any communication and coordination systems .

According to Russel et al. (2002), intelligent agent is anything that is able to perceive its environment through sensors and act through actuators upon information obtained. All agents can be categorized into 5 different groups, based on their degree of perceived intelligence and capability: 1) simple reflex, 2) model-based reflex, 3) goal-based reflex, 4) utility-based reflex and 5) learning agents. This project is only focusing on the second category, which are also known as stateful reactive agents. This group of agents store some information about their environment and percept history as their internal state, which enables them to act upon partially observable environment (Russel et al., 2002). Although stateful reactive agent model is relatively simple, it is still widely in use especially when built upon finite-state machine (FSM) architecture. FSM is comprised of finite number of states and transitions between those states (Ben-Ari et al., 2018). Each transition is taken based on some condition and agents' actions are based on their state. Figure 1 shows an example of how a stuff animal with a pre-recorded voice line, that is activated/deactivated with a button can be displayed as a state diagram.

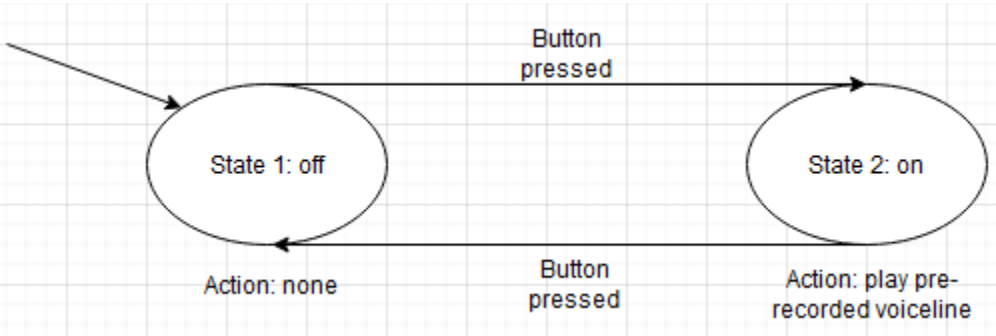


*Figure 1.* Stuff animal with pre-recorded voice line as a state machine.

Stateful reactive agents with FSM architecture are able to achieve quite complex behavior, however as noticed above, there is certain extend to which one individual agent is enough.

Multiagent systems offer a set of unique capabilities that are beyond the reach of one even a very complex robot. The simplest example is coverage – numerous simple agents are able to cover much larger distances. Additionally, there is a vast domain of tasks, that can be decomposed into a number of smaller jobs, which is a perfect use case of multiagent model, due to their ability to perform multiple, often different tasks in parallel. This model is also more reliable and scales better (Shirkhodaie et al., 2000). The main overhead of switching to multiagent systems is the need to establish proper coordination and communication among agents. There are two approaches for that matter: 1) centralized (supervised) and 2) decentralized (distributed). The first model is composed of a group of working agents and a supervisor, that assigns all the tasks and controls positioning, while the second model is comprised of independent individual robots with behavioral solutions to the multiagent environment. The centralized model is more traditional and can be taken as a ”direct extension of the traditional single-vehicle-based control philosophy and strategy” as points out Cao et al. (2012). In the same article, they acknowledge that even though both approaches have their own use case scenarios, distributed model is considered more promising due to physical constraints of centralized approach e.g., short wireless communication ranges, narrow bandwidths etc.

As disclosed earlier, this project is experimenting with vacuum cleaner robot environment, which is an application of complete coverage problem. Kong et al. (2006) suggested a distributed approach, where the space is divided into cells, each to be accommodated by one agent. Communication between agents is not restricted and assumed global, enabling an efficient coverage. In this project, we adopted the idea of cell division, however changed the communication and coordination approach to be centralized, which seems more natural given the absence of constraints on communication between robots.

Distributed systems are often based on rules around local information. This method was originally inspired from our understanding of animal behavior. For instance, Reynolds (1987) managed to achieve complex birds' behaviors such as collision avoidance, velocity matching and flock centering, all based only on information from nearby agents. However, recently much more attention has been given to swarm architectures, which are based on insects. Sharkey (2006) provided a very good overview of swarm robotics, as well as good arguments on why this approach is viable. General idea behind distributed behavior of the swarm architecture is that every agent is controlled by the sum of an attraction and a repulsion forces. Attraction is forcing an agent to stay with a group, while repulsion to keep their own territory. This idea of opposite forces driven behavior was incorporated into distributed system of this project.

**Designing environment and agents.**

*Environment.*

Environment is represented as a 1000 on 1000 grid. Dirt is randomly distributed around the map in clustered manner and every 100 iterations of the simulation a new cluster is being added, to achieve more realism. Every dirt instance that is hovered by agents is getting picked up immediately. Agents are awarded points for cleaning the litter and penalized every iteration for all uncollected dirt, thus adding speed and coverage into performance assessment of systems.

Additionally, every 500 iterations a priority area is announced to agents. Cleaning all litter within that area awards the system with significant bonus which is however penalized by the length of existence of priority zone. If the area wasn't handled by the time a new area is added, no bonus is awarded, and new zone overwrites the previous one . This factor will help assess flexibility of systems to randomly arising tasks.

*Agents.*

All agents are implemented in stateful reactive fashion with the help of FSM architecture with some enhancements. Firstly, each agent besides storing information about current state also stores a stack of states to return to. Each state contains information about current target and boundaries. The bots are developed to stay within those boundaries, regardless of any observable conditions. The actions to be taken towards the target differ based on current state. Once the task is considered accomplished by robot, the current state is being replenished by the state in the stack. There are two base states: 1) moving to the target 2) moving away from the target. Additionally, there is also a higher order state called Cleaning. If the agent is in this state, then it searches for the closest dirt and sets the current state to moving towards it, while also pushing itself onto the stack of states to return to. The search accounts the turn that has to be made to be aligned with the target, and is chosen to be strictly greedy, to minimize the loss from uncollected dirt penalty.

This implementation of agents makes it simpler to control them. High level orders like move to a specific location and clean there can be easily provided by supervisor by setting a stack of states, while implementing some additional update state functionality will enable the agents to act in a distributed way. Additionally, this design ensures that agents can act independently without proper communication and coordination, hence ensuring that baseline method doesn't have any unfair disadvantages to more complex coordinated models.

*Control models.*

As mentioned earlier both supervised and distributed models have been implemented for this project.

In centralized system, supervisor will be splitting the grid into cells, one for each agent and assign the closest robots to clean those territories. The system will handle priority zones by sending agents to clean the part of the priority area that overlaps with their assigned cell.

In distributed system, each agent will start the simulation with the task of cleaning the whole grid. They will have an extended functionality of keeping the track of priority area on their own. If the bot is inside the area they will switch their state to start cleaning it. Distributed agents, when searching for the closest dirt will also consider repulsion and attraction forces. The repulsion is based on the distance to other agents - the closer they are, the higher it is, however with some capped value. Attraction force appears, when priority area is set – the closer the dirt to priority area, the higher the force. However, following same logic as repulsion, there is some maximum and minimum value for this force, which ensures that only agents that are already near priority area will seek to get even closer, until they reach it and start clearing.

Agents in the baseline method are also aware of priority areas, and reaching it will change their state as well, however there are no additional rules to get to the area, the baseline agents are driven by greedy search for dirt only.

**Implementation challenges.**

First issues started with implementation of movement; In particular the original design of using observations of distances from two sensors when choosing direction of movement wasn't sufficient to implement complex movement strategies. This was especially noticeable when implementing avoid running out of bounds behavior. It was then decided to switch to oversimplified model of movement, where the agent is able to understand relative positioning of other objects around it and work with angles. This is happening under the assumption that the sensors are replaced by cameras and depth analysis is implemented by default. Next issue was related to movement physics simulation. In real life when some movement is taken, it happens smoothly, while in computer simulation the change is instant, which introduces cases when the agent stays at one position and keeps turning from side to side, unable to reach the perfect angle. As a solution it was decided to introduce an error margin into calculations, which enables bots to continue their movement, adjusting their direction overtime.

Although the issues above were causing some disturbance, the most challenging aspect of development was implementation of distributed system. In particular setting repulsion and attraction formulas followed by tweaking the marginal constants. Even the slightest change would drastically affect the efficiency of the system in both negative and positive ways.

**Structure of experiments.**

There have been 3 sets of experiments, with 2, 4, and 8 agents respectively, in order to see the scaling capabilities of models. Number of moves is set to 1000 for all experiments, however initial amount of dirt increases from 2000 for the first experiment to 3000 and 4000 for the other two. For each experiment set, 10 runs have been made to get more generalized results.

Note: Numbers 2, 4, 8 do not carry any additional meaning, they have been chosen to simplify cell division functionality of supervised control model.

**Results.**

Table 1, 2 and 3 all demonstrate average results of 10 runs in all four sets of experiments. Each table is identical in its structure. First column states the number of agents that were used in this particular experiment. Second column demonstrates the overall score, which is the main parameter of performance assessment. Third and fourth columns state the amount of dirt and percentage of dirt collected to the total amount of dirt dispatched respectively. The last column demonstrates relationship of total number of moves all priority areas existed to the total number of moves the experiment continued, which gives us an insight on time required to clean it in different configurations.

**Table 1.**

*Results of experiments on Supervised control model.*

| Number of bots | Score | Dirt collected | Dirt collected (%) | Existence of priority area (%) |
|---|---|---|---|---|
| 2 | 1061.78 | 1075 | 50.67 | 66.22 |
| 4 | 2335.59 | 2376 | 75.96 | 60.60 |
| 8 | 4608.62 | 4091 | 99.44 | 52.22 |

**Table 2.**

*Results of experiments on Distributed control model.*

| Number of bots | Score | Dirt collected | Dirt collected (%) | Existence of priority area (%) |
|---|---|---|---|---|
| 2 | 873.77 | 1028 | 48.34 | 69.36 |
| 4 | 2418.35 | 2415 | 77.49 | 47.46 |
| 8 | 4659.87 | 4077 | 98.94 | 36.56 |

**Table 3.**

*Results of experiments on Baseline model.*

| Number of bots | Score | Dirt collected | Dirt collected (%) | Existence of priority area (%) |
|---|---|---|---|---|
| 2 | 890.93 | 962 | 45.42 | 80.94 |
| 4 | 2128.24 | 2353 | 75.28 | 78.28 |
| 8 | 3786.14 | 3732 | 90.49 | 60.88 |

**Discussion.**

All three tables demonstrate decent scaling of dirt collected with respect to number of bots. This is due to the fact, that robot vacuum cleaner environment is an example of complete coverage problem, which is in general a perfect use case for multiagent environment regardless of the control system. However, columns 1 and 4 are capable of providing additional insights on performances of models. Firstly, distributed system is significantly better at handling priority areas, insisting on its flexibility, which also scales with number of agents. Meanwhile supervised model, even though performs relatively well in this factor, doesn't scale as well. This is however resulted from using static cell division architecture. Increased number of agents implies more cells, which also increases the chance of multiple cells overlapping with priority area, yet not significantly, which is proven by experimental results. Even baseline method seems to be showing better scaling at large number of agents, while losing at every other stat. Despite of having much better performance in handling priority area and approximately same amount of dirt collected Supervised model demonstrates almost identical results as distributed one. This implies

that supervised system collected a significant amount of dirt at the beginning, resulting in less average penalty for uncollected dirt overtime. This comes from very good initial coverage of static cell division. However, over time speed was slowing down, which may be observed by approximately same results in dirt collected factor, which is also natural for this architecture, since agents that finished their cells are not being in use until the end of the run, while other models use all their agents. Concluding observations: current implementation of supervised system has very good initial coverage, yet bad flexibility, and performance worsens overtime while distributed system is showing non exceptional, but stable results –none of these two is better on average, each has its use cases based on their advantages. Unfortunately, baseline model demonstrates much worse performance, especially in terms of score scaling. This comes from the fact that agents don't communicate or coordinate and often have target overlapping, which only increases with number of agents, hence with increase in required agents for the given problem, even if the problem is simplistic, it is more sensical to bare additional cost of implementing control system, that will provide the means for communication and coordination.

**Conclusion**

In this project we managed to implement supervised and distributed control systems for multiagent vacuum cleaner environment. Those systems have been then assessed on their coverage, flexibility, and scalability through a set of experiments, where both systems were compared to a third baseline method which doesn't use any means of coordination between agents, to create better ground for discussion. Although initial questions have been answered, implemented agents and environment run under assumption, that their sensors (cameras) are able to perfectly identify litter, which can't be guaranteed in real life applications. Additionally, distributed system was built incorporating ideas of repulsion/attraction forces driven behavior, however the movement of agents to the target was implemented to be stateless – repulsion doesn't affect the bot that is on its way to the target, which occasionally results in target overlap between agents. Last but not least, supervised system was built upon static cell division method, which is known to provide high coverage, however, is very inflexible by design, thus makes scoring system, which is heavily impacted by flexibility (high reward for cleaning priority area) - biased. In future works we hope to address those issues by redesigning the agents themselves and improving the logic behind the control models. Additionally, we would like to implement AI driven control systems and compare them to the current ones.

References.

Russell, S., & Norvig, P. (2002). Artificial intelligence: a modern approach.

Ben-Ari, M., & Mondada, F. (2018). Finite state machines. In *Elements of Robotics* (pp. 55-61). Springer, Cham.

Shirkhodaie, A., Goetz, R. C., Patkar, A., Masour, M., Aderogba, S., & Avalareddy, N. (2000, July). Behavior-based tactical navigational and mobility control of tandem ground robotic vehicles. In *Unmanned Ground Vehicle Technology II* (Vol. 4024, pp. 44-54). SPIE.

Cao, Y., Yu, W., Ren, W., & Chen, G. (2012). An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial informatics*, *9*(1), 427-438.

Kong, C. S., Peng, N. A., & Rekleitis, I. (2006, May). Distributed coverage with multi-robot system. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* (pp. 2423-2429). IEEE.

Reynolds, C. W. (1987, August). Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (pp. 25-34).

Sharkey, A. J. (2006). Robots, insects and swarm intelligence. *Artificial Intelligence Review*, *26*(4), 255-268.