# Firefighting Drone

The firefighting drones operate as follows:

* The firefighting drones start once the initial survey of the forest has been carried out.
* They will hover in the middle of all the survey drones.
  + Using K-Means Clustering
  + This will allow them to be as close as possible to all drones
* Once survey drones have identified a fire, firefighting drones will have a bidding war based on:
  + Distance to fire
  + Battery remaining
  + Water levels
* The firefighting drones will then go to the fire and drop its payload (water) onto the fire.
* Once water levels are depleted, the firefighting drones will return to the central station to gather more water.
  + Once a firefighting drone returns to the central station, a new bidding war between remaining firefighting drones will occur to replace the refueling one.
* The newly filled firefighting drone will then return to the center of the cluster.

## Deductive/Reactive/Proactive:

Taken from Shivam’s notes:

Reactive:

* Reroute to another path in case of obstacles
* Adjust water suspension based on extent of fire
* Call for maintenance in case of breakdown or low battery

Proactive

* Adjust ground clearance to get more clear view of fire

Utility

* E = {e0,…, em, … en }
* Where each ei in E describes of decreasing order of extent of fire (%)
* Possible actions at each environment state include:
* Moving closer/away from target area
* Increasing/Decreasing flow of water
* ∀ ei in E, accommodating other environment factors that may lead to further increase in fire,
* P(ei+1), after applying action ai on ei , if action ai is more effective than other factors
* P(ei-1), after applying action ai on ei , if action ai is less effective than other factors
* P(ei), after applying action ai on ei , if action ai is equally effective on fire other factors
* U(ei) ∀ ei in E = 2 \* (% decrease in fire extent) – (% amount of water used)
* This leads to a tradeoff between both values to accommodate over utilization of water in case of minor fire.
* Giving (% decrease in fire extent) twice the weight to prioritize ceasing fire over saving water.

## BDI:

Beliefs:

|  |  |  |
| --- | --- | --- |
| **Belief** | **Summary** | **Predicate** |
| AvailableParts | What parts are available for the maintenance drone to replace. | AvailableParts(<List>)  #A list containing all available parts for the maintenance drone. |

Intentions:

## 

|  |  |
| --- | --- |
| **Intentions** | **Sample Actions** |
| **Constant** | |
| maintain\_ground\_clearance | measure\_distance\_from\_ground  adjust\_ground\_clearance |
| proximity\_sensing | scan\_for\_objects\_in\_close\_proximity  scan\_for\_objects\_in\_distant\_proximity |
| obstacle\_avoidance | adjust\_route\_for\_objects\_in\_close\_proximity  adjust\_route\_for\_objects\_in\_distant\_proximity |
| power\_supply\_level\_sensing | calculate\_battery\_level |
| hardware\_malfunction\_sensing | perform\_sensor\_check |
| software\_malfunction\_sensing | perform\_software\_check |
| communications\_listening | radio\_communications\_on  frequency\_scanning |
| water\_level\_sensing | get\_sensor\_reading\_on\_water\_container |
| temperature\_sensing | get\_readings\_on\_all\_sensors  calculate\_temperature\_average |
|  |  |
| **Events** | |
| initialise | check\_battery\_level  check\_water\_level  get\_incident\_location  calculate\_shortest\_path |
| put\_off\_fire | locate\_target  locate\_other\_agents  calculate\_optimal\_proximity\_to\_fire  move\_to\_optimal\_position  put\_out\_fire  check\_water\_level\_for\_refill  check\_for\_hardware\_malfunction  check\_barrery\_level\_for\_recharge |
| fill\_up\_water\_containters | check\_water\_level  calculate\_shortest\_path\_to\_water\_source |
| battery\_replacement | broadcast\_location\_and\_battery\_information\_to\_station  listen\_to\_station\_for\_battery\_change\_information  broadcast\_to\_maintenance\_drone\_precise\_location  scan\_for\_maintenance\_drone\_in\_close\_proximity  engage\_battery\_swap\_procedure (to be decided exactly later) |
| return\_to\_station | locate\_station\_location  calculate\_shortest\_path\_to\_station  go\_to\_station |

Desires:

“!φ” denotes “achieve φ”, “?φ” denotes “query φ”

|  |  |  |
| --- | --- | --- |
| Agent | Desire | Rationale |
| Firefighting (Df) | ? FireReported | Check for fire report |
|  | ! ∃(not (FireReport)) | There should be no fire if they do their job |
|  | ! StoredWater(Full) | Try to maintain full water |

Temporal Logic:

*Legend: agent\_name (listens\_for) [asks\_for]*

Df (fire\_alert, backup\_call, support\_eta) [water\_out\_alert, need\_backup, fire\_out, request\_battery, breakdown\_alert, alert\_humans]:

⌾ fire\_alert → ◇ reach alert coordinates

⌾ backup\_call → ◇ reach backup coordinates

⌾ fact(charging<50) → ◇ give(request\_battery)

⌾ fact(water\_out) → ◇ give(water\_out\_alert) AND ◇ give(need\_backup)

⌾ fact(breakdown) → ◇ give(request\_maintenance) AND ◇ give(need\_backup)

⌾ fact(code\_red)→ ◇ give(alert\_humans)

(fire\_alert U fact(fire\_ceased)) → ◇ give(fire\_out)

## Communication, coordination and cooperation:

Communication occurs when a firefighting drones leaves the location of a fire. It will invoke a communication to the rest of the firefighting drones that a replacement is required. It will communicate the following:

Drone ID, Fire Location

Communication can also occur when a firefighting drone makes a request for new parts to a maintenance drone.

## Negotiation:

When a bidding war occurs, each firefighting drones produces a “score” heuristic, based on the following values:

Score = (water\_level + fuel\_level) - distance\_to\_fire

This score will be used in bidding wars to determine which firefighting drones go to the fires.

## Group Decisions:

N/A

## Allocating Scarce Resources:

The bidding process, in order to conserve resources will involve an inverse-synchronous English Auction. This works as follows:

* Once a fire has been identified, and a number of firefighting drones required has been sent out, a bidding process will occur, where the above mentioned heuristic will be used to determine which drones are sent.
* An English auction works as prices are increased until the highest bidder wins.
  + We inverse this, by making the lowest bidder win.
  + This is also synchronous, as all drones submit their bids at the same time.
* The lowest n bids (where n is the number of firefighters requested) are sent to the fire.