General loop for BDI model:

We distinguish two main phases in the simulation:

1. Explore the environment (find depots and the shoreline);
2. Build the embankment along the shoreline;

Initialize BDI

We assume that the first desire is to “explore the world”.

* Beliefs:
  + Global beliefs initially given to all agents:
    - # of depots in the environment;
    - # of patches that make up the shoreline;
    - # number of agents in environment;
    - Direction of the shoreline, in our case (currently) always a vertical direction (this knowledge can be used when exploring the shoreline) and “we know” that the “sea” is in the East of our world (which should be also referred to as global knowledge);
  + Location of depots: empty list;
  + Patches that make up the shoreline: empty list.

Note: It would be helpful to sort the patches based on their y-coordinate because we belief that the shoreline has a vertical direction (that’s how we implemented is currently). Sorting it on y-coordinate will make a meaning ordering that can be used for exploration and construction.

* + # of agents at the shoreline. We could use this information to possibly optimize the exploration or the start of the building process, because you could imagine that it is not optimal that they are all heading to the shoreline, if some of them found pieces they can probably explore the rest of the shoreline quite efficiently. We could use a certain threshold, if more than 6 agents are already exploring the shoreline “do something else”. E.g. search for depots. If the location of all depots is already found, then go to one of them (in your neighborhood) and wait there until the complete shoreline is explored. We can then experiment with the “threshold” for example and see what is optimal.

* Desires:
  + “Explore the world”
* Intentions:
  + “explore randomly”

Note, because agents will communicate with each other via different types of messages, each of them will have (for now I will assume that they will be all separate objects, but possibly we can find a way to find a more generic implementation where we have one queue and the message has a “header/type”, for me that has low priority):

* A message queue (list) to communicate the locations of the shoreline;
* A message queue to communicate the locations of the depots:
* An idea for later: for optimization purposes it could be an idea that an agent sends a message where its current location at the shoreline is (for those that are there, having found a piece). This information could be used a) to determine whether it is still suitable to go in the direction of the shoreline for exploration or “do something else”. Low priority.

While the whole shoreline is not fully explored or not all depots have been located

* Perceive:
  + Use your vision radius and look for:
    - Water
      * Perceiving a “patch of water” is not enough to deduce where exactly the shoreline is situated because the bumpiness of the shoreline also requires to know where the next “water patches” adjacent to this one are located (easy to see when you look at the actual coastline). But I propose the following:

1. Postpone the reasoning to the very end when all patches have been observed;
2. And for complexity “minimization” we could also say that we start with a shoreline that is a straight vertical line (my preference);
   * + Depots

* Update your beliefs:
  + Note, if you look at “other” BDI models, they always have the sequence of “first perceive and then update your beliefs”. We did not do that in our previous implementations but I think it makes more sense this way.
  + If shoreline is not complete: update beliefs about location of shoreline:
    - Because “we know” that the shoreline has a vertical direction, we can always deduce that the patch “to the left of the observed water patch” must be a “shoreline patch”.
    - If that patch is not already part of your beliefs, add it to your beliefs.
  + If not all depots already located: update beliefs about location of depots:
    - If location not yet part of the beliefs about depot locations then add to beliefs;
  + Check your message queues for new information and update your beliefs based on that information;
* Communicate new beliefs:
  + If the agent has discovered new beliefs (e.g. about the location of a depot) share that information with the rest of the agents by sending a message (see above for the different message objects);
* Update desires
  + As long as both locations (depots, shoreline) are not fully explored equal to “explore the world”.
* Update intentions (ONLY IF DESIRE IS EQUAL TO EXPLORE THE WORLD)

Defining the following events for readability:

Events:

All(S): found all shoreline patches

SOME(S): some of the shoreline patches were found already

All(D): found all depots

FS: found a patch (at TICK - 1)

AT\_S: positioned at shoreline (one of the neighborhood patches is “water”)

IF NOT ALL(S) AND NOT ALL(D)

IF FS AND NOT ALL(S)

/\* Distinguish two situations, agent is at the shoreline of close to it (in vision radius) \*/

IF AT\_S

Intention = “move along shoreline” (to be defined in action)

OTHERWISE

Intention = “move to shoreline” (to be define in action);

ELSEIF ALL(D) AND SOME(S)

Intention = “move towards explored shoreline”

OTHERWISE

Intention = “explore world randomly”

Extension for later (or never), if we know how many agents are currently “somewhere along the shoreline” because they discovered “a piece” than we could implement a threshold saying “search for depots” or if all depots have been found “go to one of the depots and wait there until the others have found all pieces of shoreline”, this would be a kind of preparation step for the “construction phase;

* Execute actions

Explore world randomly

Move to one of the patches in your “neighborhood”:

No other agent on that patch;

No depot on that patch;

Patch must be sand-terrain (we defined already a variable “terrain-color”);

Move along shoreline

1. The simple case, assuming the shoreline is a straight, vertical line:
   1. Hold the x-direction;
   2. Can you move in both directions (x,y+1) and (x,y-1)?

Is patch (x, y+1) and (x, y-1) already discovered?

Randomly choose one y-direction

Otherwise

Choose the y-direction that is not yet discovered or choose randomly

Otherwise

Choose the y-direction you can move-to

1. Complex case, the shoreline has a zigzag form (bumpy):

Note: not considering the vision radius, otherwise it is getting too complex.

If FS (found water at tick-1)

Is patch (x, y+1) and (x, y-1) already discovered?

Choose y-direction randomly

Otherwise

Choose the y-direction that is not yet discovered or choose randomly

Otherwise

Agent must have already a preferred y-direction

Preferred y-direction North (y+1):

Goal, make sure you always have water on the right side of you when going North

You can face North, West or East

If neighbor patch on the right is “water” and can move forward

Forward 1

ElseIF neighbor on the right is “water” but can’t move forward

Turn left

ElseIF neighbor on the right is NOT “water”

Turn right

Otherwise

Choose randomly a direction where you can move-to

and otherwise to the left

Move to shoreline (both cases straight vertical and zigzag shoreline)

1. Go in the (x,y) direction of the shoreline patch last found.