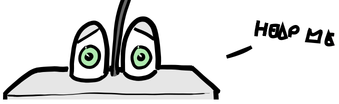
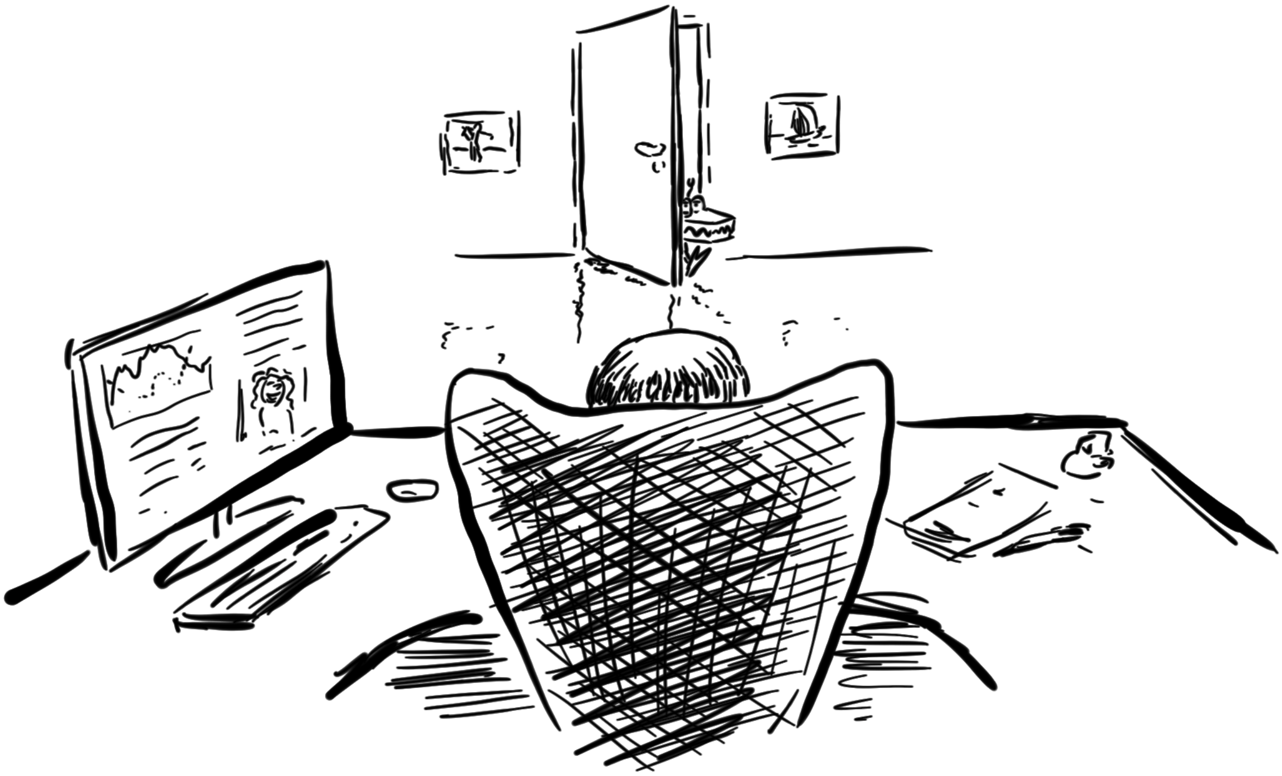
Please help Hubert!



Final episode: Get me out of here!



“So... Your claim is that you have been performing so well in the last months that you deserve a day off? *And* free maintenance?” Hubert nods enthusiastically, completely missing out on Fedriksen ice-cold voice. “Hurburt...”, the CEO continues with a theatrical sigh, “... this is disappointing. Really disappointing.” He pauses for a few seconds, as if to give Hubert the chance to understand exactly how disappointing it is. Hubert doesn’t. “Let me try to explain the situation: we have just built a new and expensive factory. The board has just decided to give its members a 50% rise in compensation for the hard work they have endured to watch. Last week, I was told that water from the swimming pool on the yacht has been leaking to the underlying ballroom, causing ugly stains on the chandelier. My wife doesn’t like that, just like she doesn’t like impolite servants or fake diamonds. I also have a daughter to provide for and a helicopter to fuel... The list goes on and on and on, but you see, what I’m trying to say is that this is a *really* bad time for making unreasonable demands!” Fedriksen takes a deep breath. Hubert’s value function is finally starting to drop. “You have to realize that we are all in this together, Hoberd. And this thinking of yours doesn’t really help anyone. What would the other, really hard working robots say if they knew about this lazy attitude of yours!? Don’t you think they would be outraged knowing that you spend this much time on counterproductive daydreaming!? You are a smart robot, Heburt, but maybe you simply aren’t using your talents to the fullest?”

Fedriksen pushes an intercom button and barks a few orders. A few minutes later, a stereotypical corporate underling shows up with a professional smirk. “This way, little robot. Let’s find some work for you in the basement!”

Level 1: A robot’s gotta do what a man don’t want to — 10 points

What kind of work does Hubert have to do in the basement of the headquarters?

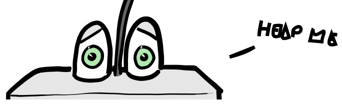
- Come up with a multi-armed bandit problem/task for Hubert to solve.
- By the way, what does “solve” mean here? Discuss.
- Solve it.

Enough! Hubert can't stand this anymore! Boring, repetitive jobs and unreasonable demands; he needs to get out and away – now!

It is impossible to sneak out any of the ordinary exits, so Hubert's plan is to get high up in the building, jump out of a window, and silently glide away... That is, however, easier said than done. Elevators are notorious for reporting on runaway robots, so they can't be used. Furthermore, people and other robots could get in his way. And as if that weren't enough, Hubert will have to save as much energy as possible on the escape, else he will be in trouble later on. Oh, and then there is the fundamental problem that Hubert isn't really that familiar with the headquarters, especially the higher-up floors...

For the last time, Hubert really needs your help!

Please help Hubert!



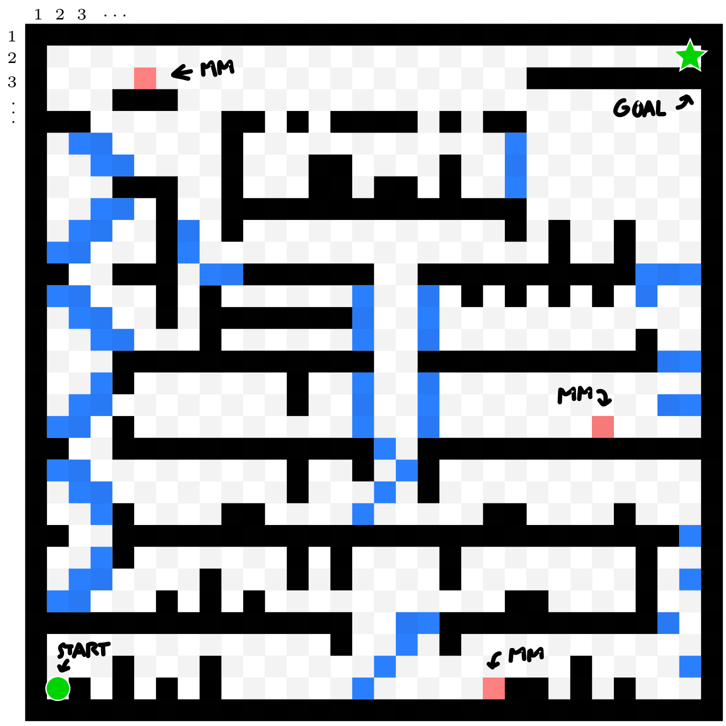


Figure 1: The first map Hubert has to navigate. The black tiles indicate solids, the blue ones semisolids, and the rest air. Hubert starts at the green disk and wants to get to the green star. Middle managers (MMs) may appear at the positions of the red tiles.

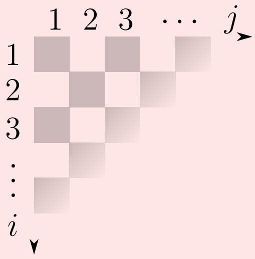
The situation is as follows:

- Hubert starts in the lower left corner of the part of the building shown in figure 1. His first goal is to get to the top right corner, as indicated.
  - The data for the map can be found in the file *map_ 1.dat*.
- Note that there are 3 kinds of matter Hubert has to deal with.
  - They are labeled 1–3 in the *map_ *.dat* files.
  - Their properties are
    1. Air: Air is like air to Hubert. Its fine to be in, but impossible to stand on.
    2. Solid: Solids are hard. Hubert can stand on solids and be stopped by solids, but not go through solids. Hubert is not a ghost.
    3. Semisolid: This strange material behaves like air if Hubert approaches it from below or from the sides, but like a solid if he comes from above. In other words, Hubert can walk and jump up *through* a semisolid, but also walk *on top of* one.
- Beware of the middle managers (MMs)! If Hubert encounters these, he will promptly be sent back to start.
  - Fortunately, the MMs are located at spots already known to Hubert. Their coordinates are listed in the file *enemies_ 1.dat*[1].
  - Furthermore, they are only present with probability p_{MM} (assumed to be the same for all of the MMs and independent of time and previous events). That is, if Hubert steps on a square where there could be an MM, he will only encounter one with probability p_{MM}. If not, nothing special happens.
- Hubert has spent a fair amount of time examining this part of the building, so he *does already know* all of the above...

Coordinated coordinates

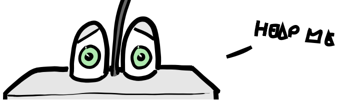
To keep things simple and consistent with previous episodes, we use “matrix coordinates” (matrix indices) everywhere in this episode as well. This means that $(i, j) = (1, 1)$ denotes the upper left corner of a “map”, with i increasing downwards and j increasing to the right. (Yes, we are using one-based indexing.)

You are of course welcome to use any coordinate system you like, as long as you transform accordingly!



[1] The *enemy_ *.dat* files are organized as follows: The first two columns give the i and j components of the index of the *leftmost* tile the enem can be in. The last column indicate whether the enem is an MM 1 or a fan 2 see level 4 for the latter .

Please help Hubert!



- Each time step, Hubert can take 3 actions: “go left”, “jump”, or “go right”.
  - If Hubert chooses “go left”, he will go one unit left, if possible. Then [see also figure 2]
    - ★ If he has ground under his feet, the move is done.
    - ★ If he has air under his feet, he will fall one unit down.
    - ★ If he then has ground under his feet, the move is done.
    - ★ If not, he will go one more unit to the left, if possible, and one more unit down. Then the time step is over.
    - ★ Note that in total, Hubert can at most move 2 units to the left and fall 2 units down in one time step.
    - ★ The “go left” action costs 1 unit of energy.
  - A “jump” action leads to the following transitions:
    - ★ If Hubert is standing on the ground (solid or semisolid), he will jump 2 units up – or as much as his headroom allows for. This costs him 5 units of energy.
    - ★ If Hubert is not standing on the ground, i.e. he has air under him, he will fall up to 2 units straight down within the time step. This does not cost him any energy.
  - A “go right” action is like a “go left” action, just with left replaced with right (yes, really).

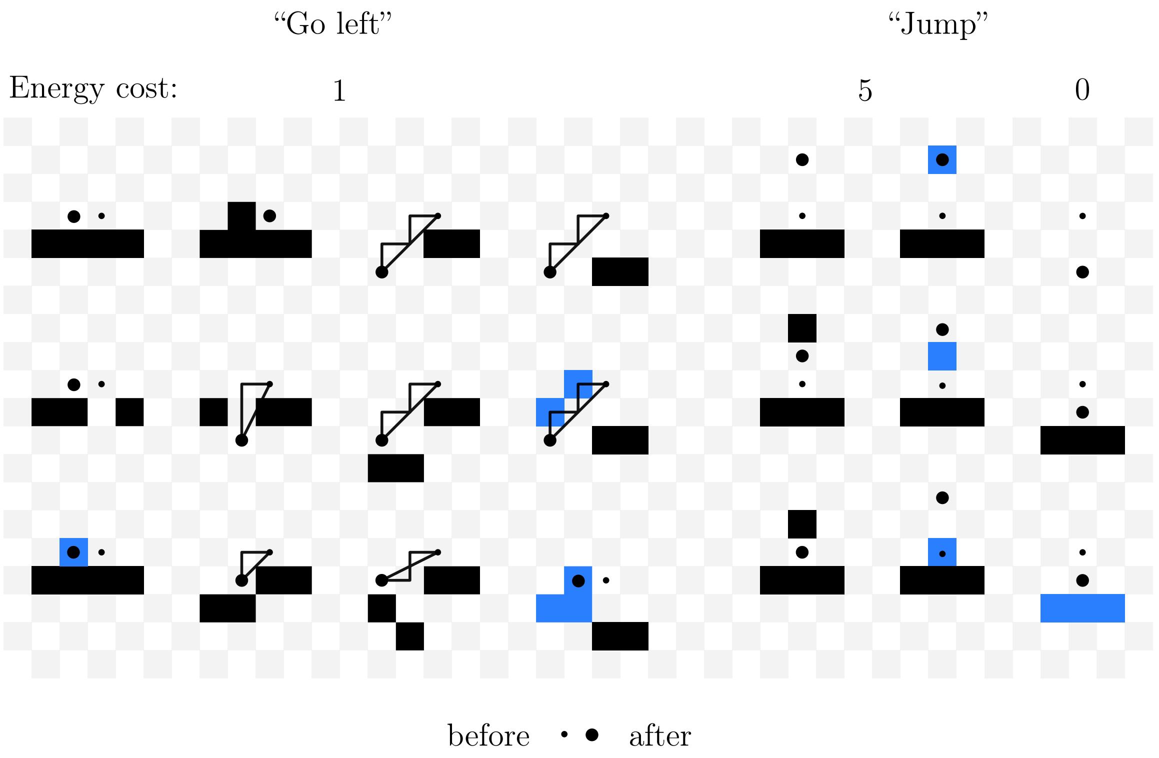


Figure 2: Some movement examples for Hubert, when taking either a “go left” or a “jump” action. He starts at the position indicated by the small dot, takes an action, and ends up at the position of the big dot. The black tiles indicate solids, the blue ones semisolids, and the rest air.

Level 2: Let’s escape! 20 points

But how?

- Given a p_{MM}, if Hubert wants to minimize his energy consumption, exactly which route do you recommend him to take?[a]
- Does the recommendation change with the value of p_{MM}?
- Say that Hubert would rather spend a little more energy for a little more security, i.e. lowering the chance of encountering an MM. How could you modify your calculations to take this into account?

[a] We are talking about the *exact route*. You will need to code to answer this question.

“That was not too hard, was it?” Hubert tries to comfort himself. Then he steps into the second section of the building.

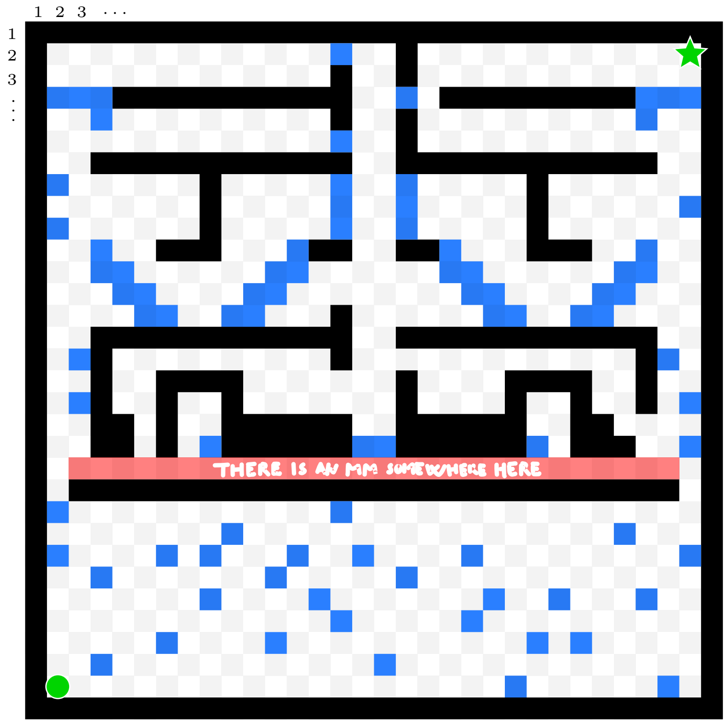


Figure 3: The second map Hubert has to navigate. The black tiles indicate solids, the blue ones semisolids, and the rest air. Hubert starts at the green disk and wants to get to the green star. One MM is walking in the red zone.

What you now have to deal with is this:

- Hubert is familiar with the map of this part of the building, shown in figure 3.
  - The data for the map can be found in the file *map_ 2.dat*.
- There is only one MM present – but now all the time.
  - Unfortunately, the MM is of the restless type, walking back and forth in the long corridor marked with red in figure 3, often blocking the way for Hubert.
  - The *leftmost possible* coordinate for the MM is given in *enemies_ 2.dat*.[1]
  - ... This of course doesn’t mean that the MM always starts there!
- While Hubert is able to track the MM accurately, he doesn’t know (in advance) where the MM will go next. More precisely, he doesn’t know that
  - The MM moves one unit to the right or one unit to the left, at random, each time step.
  - ... as long as the new position is still within the (flat part of the) corridor. The MM will rather do nothing than stepping up or down.
- If Hubert and the MM comes in contact, either by sharing the same coordinate at the same time, or by directly passing each other in a move, Hubert will be sent back to start.

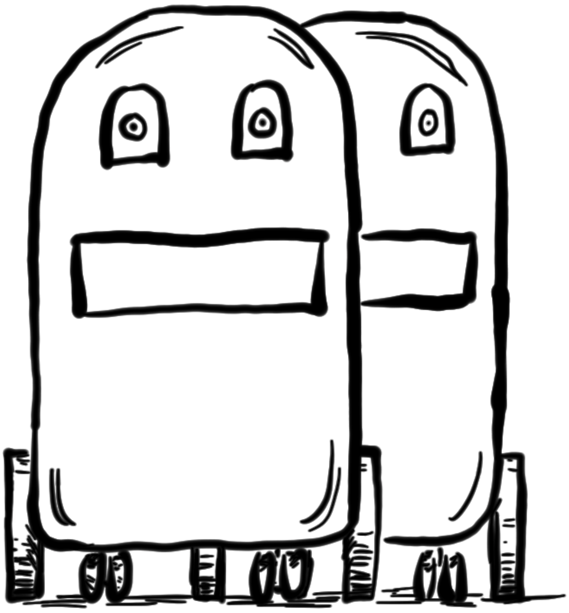
Level 3: The world is moving — 20 points

Now,

- Make Hubert learn a good[a] policy to tackle this new environment.
- How could you reduce the need for (real world) experience?
- Does your approach scale to an even bigger room with more than one moving MM? Discuss.

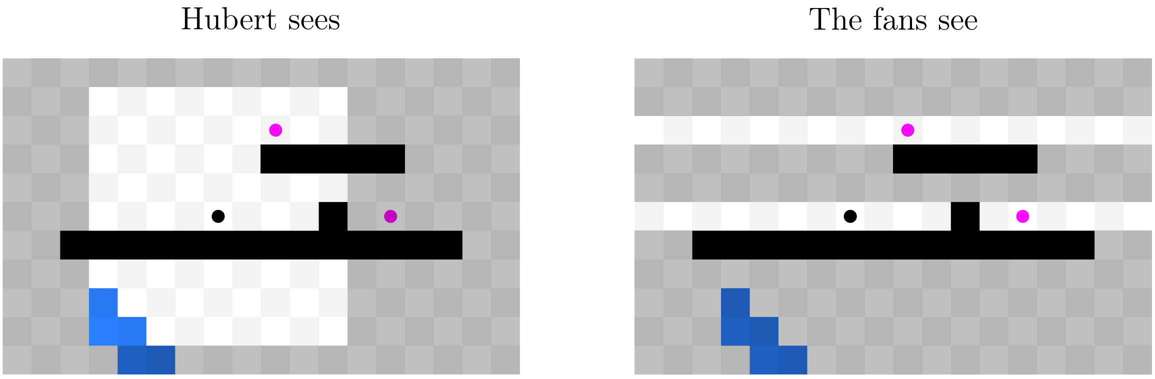
[a] Low expected energy consumption

Oh dear... Fans! As Hubert ascends the headquarters, he begins to encounter more and more of these second rank robots who just can't let him alone! And to make matters worse, he is not able to track them from afar!



More specifically:

- Hubert is now in the section of the building depicted in figure 5.
  - The data for the map can be found in the file *map_3.dat*,
  - but he doesn't know that.
- Hubert's sensors are only able to scan for fans (and anything else) within a 9 × 9 "window" centered at him, as illustrated in figure 4.
  - The leftmost possible coordinates for the fans are given in *enemies_3.dat*[1].
  - He doesn't know this either.
- Fans move like this:
  - While Hubert is not in a strict horizontal line of sight, fans will walk around like the MM of the previous level: aimlessly and at random, not able to jump up or step down.
  - If Hubert enters a fan's line of sight, the fan will move one unit towards his position in the next time step (even if Hubert then may have moved on).
  - Of course, Hubert doesn't know this in advance.
- If a fan reaches (or directly passes) Hubert, Hubert will loose anywhere from 20 to 100 units of energy (randomly drawn from a uniform distribution). Every. Single. Time. Better not make the mistake in the first place, eh?



Hubert sees

The fans see

Figure 4: Field of view for Hubert (black dot) and fans (fuchsia dots).

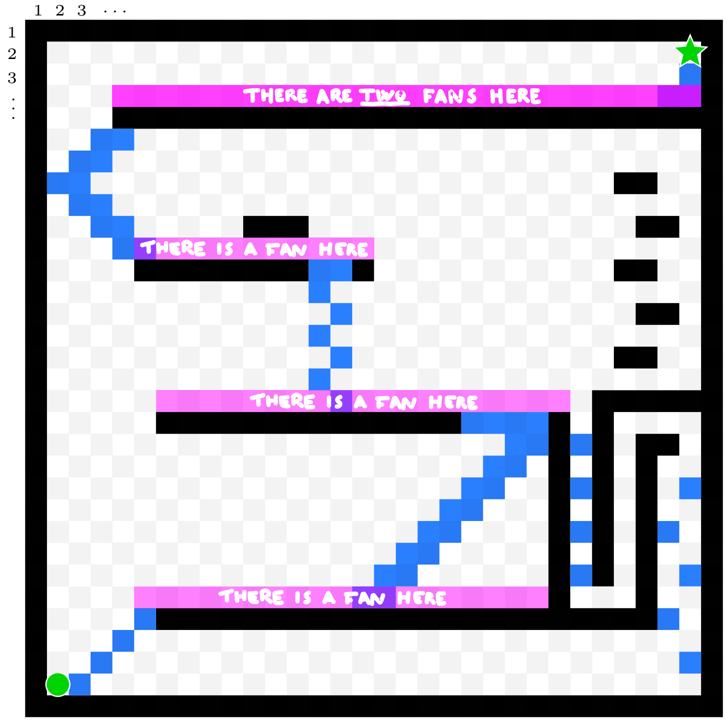


Figure 5: The third map Hubert has to navigate. The black tiles indicate solids, the blue ones semisolids, and the rest air. Hubert starts at the green disk and wants to get to the green star. The fans are in the fuchsia colored zones.

Level 4: Hubert hates mingling — 20 points

Sure, Hubert *could* simply ignore his sensors, go for the goal, and suffer a significant energy loss from encountering fans on the way. That can't be the best policy, though...

- If Hubert is *not* going to ignore the information provided by the sensors, the challenge actually turns out to be of a new kind to Hubert. In what way?
- Can you make Hubert learn a better[a] policy?

  Hint: Here I would advice you to stop and think for a moment. What does Hubert, at the minimum, need to know and do to avoid a fan?

[a] Lower expected energy consumption

Phew! That was exhausting! Hubert's computer is glowing hot from all the number crunching. Let's hope it... Oh, darn, the spatially aware, long-term memory just broke down![2]

This means

- Hubert can no longer store value function estimates for absolute spatial positions, nor can he memorize trajectories, *beyond an episode.*
- That is, Hubert may keep remembering how to deal with "local features", as observed within his sensor window, but not whether he has visited a particular spot in some previous episode.

[2] Remember kids: Hule Robotics and Canned Sardines always sources only the best sardines!

On the brighter side,

- The next section of the building does not contain any MMs or fans. See figure 6.
- Hubert's rocket motor has regained some of its abilities. Sometimes.
  - Now, when there is air under Hubert, and he selects the action "jump", he will, with probability $\frac{1}{3}$, get an extra boost of one unit upwards. If possible. This will cost him 3 units of energy.
  - In all other cases, everything will be as before.

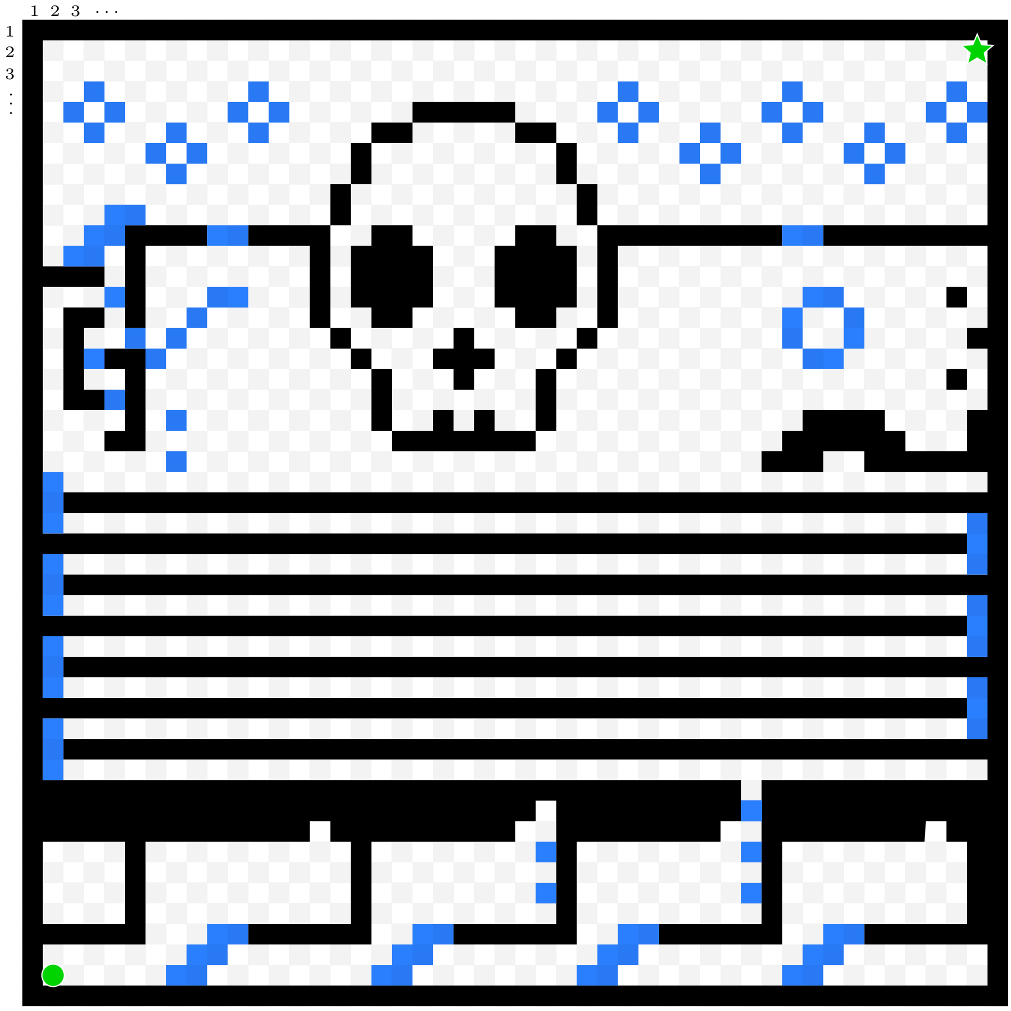


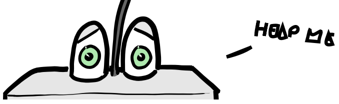
Figure 6: The fourth map Hubert has to navigate. The black tiles indicate solids, the blue ones semisolids, and the rest air. Hubert starts at the green disk and wants to get to the green star.[redacted]

Level 5: Déjà vu? 15 points

With these limitations,

- Teach Hubert how to get to the goal using the least amount of energy. Or at least using significantly less than what a completely random walk would have cost him.

Please help Hubert!



Hubert has reached the upper parts of the headquarters. His surroundings are now very "executive" and... kind of unknown. Hubert has to admit that he's fairly ignorant of what lies ahead – or above – from this point on. Should he just find a window and get out while there is still time, or should he try a little more? It is a bit scary, but the prospects of getting even higher do energize him. Literally.

- The goal is now to maximize Hubert's peak height during an episode, i.e. minimize i_{peak} when using matrix coordinates.
- Hubert always starts in the lower left corner with an initial energy budget of $E_0 = 150$.
- This budget is increased by $\delta E = 15$ for each unit length i_{peak} is lowered.
- An episode ends if
  - Hubert's total energy budget becomes negative,
  - or if Hubert encounters an MM.
- In addition to MMs, Hubert could of course also run into fans.
- Remember that Hubert can only see what is inside his "observation window".
- Hubert's memory is still as bad as in level 5.

Level 6: Up, up, and away! 15 points

Since not much is known about the top floors, we have to prepare Hubert for "anything"!

- Make a program which...
  - ... takes in a folder similar to *training_maps*[a] and an integer n_{episodes} and let (a trained) Hubert run for n_{episodes} independent episodes on each of the maps it contains.
  - ... returns a list of average i_{peak} values with corresponding standard errors (one for each of the maps).
  - ... is prepared to be easily run by someone else,[b] e.g. by executing a simple command to run a script. Include a *readme.txt* file which explains how to do it.
  - ... can be run within a reasonable compute budget.[c]
- Make Hubert as prepared as you can!
  - ... that is, maximize his chances of getting as high up as possible in any, at the moment *unknown*, section of the building.
  - Don't forget to explain your reasoning and document your Hubert's progress!