State diagram rationalisations

Dead end:

Each dead end has been given its own family of algorithms for a reverse. Since a U-turn is the easiest the algorithm will first see if there is a space to the right (it’s assumed the left-hand algorithm has been halted) and execute it if so. If this does not work, the algorithm will check if the bottom right hand side (south east where north is the car’s direction) is available for a three-point turn. It only checks for the right side since if the algorithm has reached this stage then clearly the bottom left (south-west) is not available. This has priority over straight reversing since reversing comes with dirty problems. If both cases fail, then the car will reverse all the way to the back wall. The reason for this is because if it reversed until the first side wall was available then in some instances an L-shaped dead end would create infinite cycles.

Trap:

Likewise, each trap has been given its own family of algorithms for handling it. The lava algorithm will first treat each piece of lava like a wall and mark them. If it comes back (lava must be traversed) it will accept its fate and traverse the first piece of lava. The mud algorithm will traverse the mud and then mark it. If it marks it twice, it will start to treat the mud like a wall. This is one of the few ways of creating an algorithm for the mud in a non-recursive algorithm. The grass algorithm marks it and attempts to avoid it by straight reversing. If it reaches the grass again it will just traverse it as normal. Grass can definitely be used to keep a car in a cycle so with full reversing the car has a chance to find an alternative path before being forced onto it.