**Running the Reactive Agent:** The “StateAgent” folder should replace the demo Krislet folder. The agent is executed identically to Krislet (i.e., “java Krislet”). An agent’s behaviors and next states can be changed without recompiling the code by editing AgentSpec.txt.

**Environment Model:** For the agent, the ball is either “visible”, “directly in front”, or “not visible” and is either “far”, “close”, or “unknown” in terms of proximity to the agent. Likewise, the agent’s team and opponent’s goals are either “visible” or “not visible”.

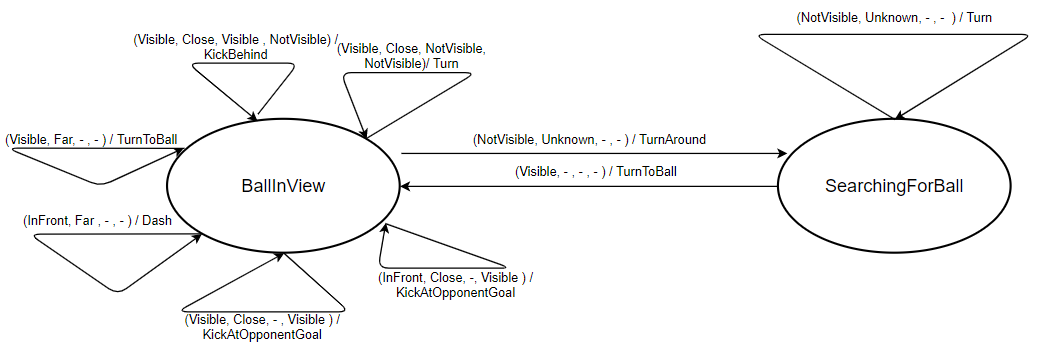
**Agent Specification Syntax:** An agent behavior is specified in the following form:

(<ball visibility>, <ball distance>, <team goal visibility>, <opponent goal visibility>) -> <action> X <state>

Where <ball visibility> {Visible, NotVisible, DirectlyInFront}, <ball distance> {Close, Far, Unknown}, <team goal visibility> {Visible, NotVisible}, <opponent goal visibility> {Visible, NotVisible}, <action> {Turn, TurnToBall, TurnToMyGoal, TurnToOpponentGoal, TurnAround, KickForward, KickBehind, KickAtOpponentGoal, Dash}, and state {BallInView, SearchingForBall}. If a behavior is not specified for a given environment the agent finds itself in, no action will be performed.

**Expected Behavior of the Provided Agent:** The agent will turn until it is in line with the ball, then dash towards the ball. The agent will the kick the ball towards the opponent’s net when it is close enough. This agent will also kick the ball backwards if it reaches the ball and can only see its own goal, and begins searching for the ball by doing a complete 180-degree spin as observations of the reactive agent showed this was most often where the ball was when an agent lost sight of it.

**State-Based Behavior:** The following figure is a state machine diagram of the provided agent specification. In order to reduce the number of transitions drawn, a – is used to denote that any value for that aspect of the environment will result in the same transition:



We can see that the behavior is in fact state-based by observing how the agent searches for the ball when it is no longer visible. In the BallInView state, an environment such as (NotVisible, Unknown, NotVisbile, NotVisible) maps to the TurnAround action. If the agent is still in this same environment state after this action is performed, it instead performs the Turn action.

**Agent Design**: A state pattern is implemented in which each concrete state has a HashMap to map environment states to actions, and environment states to next states. Each concrete state is a singleton. On each tick, the environment is discretized and used as a key to obtain the action to perform and the next state to fetch.