In your report, mention what you see in the agent's behavior. Does it eventually make it to the target location?

Yes, given enough time random movements will eventually lead the cab to its destination (or any point on our grid for that matter). Although in our scenario, even neglecting the deadline, we still hit the hard time limit of deadline + 100 iterations occasionally.

Justify why you picked these set of states, and how they model the agent and its environment.

I'm using the status of the traffic light, whether there's traffic or not, and the next_waypoint as my states. Those three items represent the most critical (in my opinion) items for determining what the cab should do next. I've chosen to skip some items in my state- namely the deadline and traffic from the left and right. Lowering the overall number of items in the state helps to reduce the overall state space and thus speed calculations/ updates which is one of the reasons I skipped deadline. Arguably, the traffic from the right and left can be mitigated by driving rules in the US, if there are vehicles on either the right or left and you have a green light your vehicle will eventually make it through the light, i.e. you can turn left on a green light after yielding to only oncoming traffic.

What changes do you notice in the agent's behavior?
It's much more intelligent now, it 'chooses' actions better and makes its way to the destination more efficiently. It also continues to improve as we iterate, thus learning along the way.

Report what changes you made to your basic implementation of Q-Learning to achieve the final version of the agent. How well does it perform?
First I implemented some reporting features to be able not identify how well the Q-Learning algorithm was doing.

I also varied Q, Gamma, and Alpha values by independently running several iterations for a given value, averaging those results, and then creating a table with all of the values. Because I wanted to maximize the success rate as a priority over net reward or penalty rate, I settled on on Q = 3, gamma = 0.45, and alpha = 0.2. You can see the rest of my results in the table included below.

Comparison of Q, Gamma, & Alpha Values

Q	3.000	Q	3.000	Q	3.000
Gamma	0.450	Gamma	0.300	Gamma	0.450
Alpha	0.200	Alpha	0.200	Alpha	0.100
success rate	0.983	success rate	0.970	success rate	0.967
Net Reward	2381.667	Net Reward	2439.833	Net Reward	2488.167
Penalty Rate	0.239	Penalty Rate	0.232	Penalty Rate	0.239
Q	4.000	Q	3.000	Q	3.000
Gamma	0.450	Gamma	0.500	Gamma	0.450
Alpha	0.200	Alpha	0.200	Alpha	0.300
success rate	0.967	success rate	0.983	success rate	0.973
Net Reward	2496.333	Net Reward	2386.667	Net Reward	2411.167
Penalty Rate	0.246	Penalty Rate	0.224	Penalty Rate	0.236
Q	2.000	Q	3.000	Q	3.000
Gamma	0.450	Gamma	0.400	Gamma	0.400
Alpha	0.200	Alpha	0.200	Alpha	0.300
success rate	0.970	success rate	0.970	success rate	0.977
Net Reward	2490.833	Net Reward	2464.167	Net Reward	2341.833
Penalty Rate	0.233	Penalty Rate	0.234	Penalty Rate	0.245

Does your agent get close to finding an optimal policy, i.e. reach the destination in the minimum possible time, and not incur any penalties?

Yes- my agent consistently improves and while not perfect increases its probability of reaching the goal destination in the minimum possible time. Looking through the last 10 trials of a run, we can see that the agent typically acts optimally. The one exception (and it is in bold text below) is the situation where the light is red, there is no oncoming traffic, and the agent selects left, running the red light and incurring a penalty. There are a few circumstances where a right turn on red also is occurring (and those are also bolded below) incurring a penalty.

Simulator.run(): Trial 89
Environment.reset(): Trial set up with start = (1, 4), destination = (7, 1), deadline = 45
RoutePlanner.route_to(): destination = (7, 1)
LearningAgent.update(): deadline = 45, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 44, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 43, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 42, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0

LearningAgent.update(): deadline = 41, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 40, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 39, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 38, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 37, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
LearningAgent.update(): deadline = 36, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': 'left'}, action = forward, reward = 2.0
LearningAgent.update(): deadline = 35, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 34, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 33, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 32, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 31, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 30, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 29, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 28, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': 'left'}, action = right, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 89/90 of 1 trials (net reward: 2091.0)
penalties/moves (penalty rate): 238/922 (0.26)
LearningAgent.update(): deadline = 27, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 12.0
Simulator.run(): Trial 90
Environment.reset(): Trial set up with start = (3, 1), destination = (5, 5), deadline = 30
RoutePlanner.route_to(): destination = (5, 5)
LearningAgent.update(): deadline = 30, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 29, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 28, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
Environment.act(): Primary agent has reached destination!
success/total = 90/91 of 1 trials (net reward: 2106.5)
penalties/moves (penalty rate): 239/926 (0.26)

LearningAgent.update(): deadline = 27, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 12.0
Simulator.run(): Trial 91
Environment.reset(): Trial set up with start = (6, 2), destination = (3, 5), deadline = 30
RoutePlanner.route_to(): destination = (3, 5)
LearningAgent.update(): deadline = 30, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 29, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 28, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 27, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 26, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 91/92 of 1 trials (net reward: 2126.0)
penalties/moves (penalty rate): 240/932 (0.26)
LearningAgent.update(): deadline = 25, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 12.0
Simulator.run(): Trial 92
Environment.reset(): Trial set up with start = (4, 1), destination = (8, 4), deadline = 35
RoutePlanner.route_to(): destination = (8, 4)
LearningAgent.update(): deadline = 35, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 34, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 33, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 32, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 31, inputs = {'light': 'red', 'oncoming': 'left', 'right': None, 'left': None}, action = right, reward = -0.5
LearningAgent.update(): deadline = 30, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 29, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
LearningAgent.update(): deadline = 28, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 92/93 of 1 trials (net reward: 2146.5)
penalties/moves (penalty rate): 243/941 (0.26)
LearningAgent.update(): deadline = 27, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 12.0
Simulator.run(): Trial 93
Environment.reset(): Trial set up with start = (5, 1), destination = (4, 5), deadline = 25

RoutePlanner.route_to(): destination = (4, 5)
LearningAgent.update(): deadline = 25, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 24, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 23, inputs = {'light': 'green', 'oncoming': None, 'right': 'forward', 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 22, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 21, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 20, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 93/94 of 1 trials (net reward: 2168.0)
penalties/moves (penalty rate): 244/948 (0.26)
LearningAgent.update(): deadline = 19, inputs = {'light': 'red', 'oncoming': None, 'right': 'left', 'left': None}, action = left, reward = 12.0
Simulator.run(): Trial 94
Environment.reset(): Trial set up with start = (2, 5), destination = (7, 2), deadline = 40
RoutePlanner.route_to(): destination = (7, 2)
LearningAgent.update(): deadline = 40, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
LearningAgent.update(): deadline = 39, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 38, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 37, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 36, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 35, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
LearningAgent.update(): deadline = 34, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
Environment.act(): Primary agent has reached destination!
success/total = 94/95 of 1 trials (net reward: 2186.5)
penalties/moves (penalty rate): 247/956 (0.26)
LearningAgent.update(): deadline = 33, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 12.0
Simulator.run(): Trial 95
Environment.reset(): Trial set up with start = (7, 1), destination = (5, 5), deadline = 30
RoutePlanner.route_to(): destination = (5, 5)
LearningAgent.update(): deadline = 30, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0

LearningAgent.update(): deadline = 29, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 28, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 27, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 26, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 95/96 of 1 trials (net reward: 2206.0)
penalties/moves (penalty rate): 248/962 (0.26)
LearningAgent.update(): deadline = 25, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 12.0
Simulator.run(): Trial 96
Environment.reset(): Trial set up with start = (3, 6), destination = (6, 3), deadline = 30
RoutePlanner.route_to(): destination = (6, 3)
LearningAgent.update(): deadline = 30, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 29, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 28, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
LearningAgent.update(): deadline = 27, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 26, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 96/97 of 1 trials (net reward: 2223.0)
penalties/moves (penalty rate): 250/968 (0.26)
LearningAgent.update(): deadline = 25, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 12.0
Simulator.run(): Trial 97
Environment.reset(): Trial set up with start = (8, 4), destination = (6, 1), deadline = 25
RoutePlanner.route_to(): destination = (6, 1)
LearningAgent.update(): deadline = 25, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 24, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 23, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 22, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': 'forward'}, action = right, reward = 2.0
LearningAgent.update(): deadline = 21, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 20, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0

LearningAgent.update(): deadline = 19, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 18, inputs = {'light': 'green', 'oncoming': None, 'right': 'left', 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 17, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 16, inputs = {'light': 'green', 'oncoming': None, 'right': 'forward', 'left': None}, action = right, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 97/98 of 1 trials (net reward: 2247.5)
penalties/moves (penalty rate): 253/979 (0.26)
LearningAgent.update(): deadline = 15, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 12.0
Simulator.run(): Trial 98
Environment.reset(): Trial set up with start = (1, 5), destination = (5, 1), deadline = 40
RoutePlanner.route_to(): destination = (5, 1)
LearningAgent.update(): deadline = 40, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 39, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 38, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 37, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
LearningAgent.update(): deadline = 36, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 35, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 34, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 33, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 32, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 98/99 of 1 trials (net reward: 2270.0)
penalties/moves (penalty rate): 256/989 (0.26)
LearningAgent.update(): deadline = 31, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 12.0
Simulator.run(): Trial 99
Environment.reset(): Trial set up with start = (8, 1), destination = (3, 1), deadline = 25
RoutePlanner.route_to(): destination = (3, 1)
LearningAgent.update(): deadline = 25, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 24, inputs = {'light': 'red', 'oncoming': 'left', 'right': None, 'left': None}, action = right, reward = -0.5

LearningAgent.update(): deadline = 23, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = 2.0
LearningAgent.update(): deadline = 22, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 21, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 20, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -0.5
LearningAgent.update(): deadline = 19, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 2.0
LearningAgent.update(): deadline = 18, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = 2.0
Environment.act(): Primary agent has reached destination!
success/total = 99/100 of 1 trials (net reward: 2290.5)
penalties/moves (penalty rate): 259/998 (0.26)
LearningAgent.update(): deadline = 17, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 12.0

Links

Discussion threads used during the project
https://discussions.udacity.com/t/next-state-action-pair/44902/19
https://discussions.udacity.com/t/how-to-get-started/171416/3
https://discussions.udacity.com/t/states-is-this-on-the-right-track/44273
Interesting code example that includes reporting features
https://github.com/jaycode/smartcab/blob/master/smartcab/smartcab/agent.py
Another student example of this project I referenced during coding
https://github.com/jaycode/smartcab/blob/master/smartcab/smartcab/agent.py
Q-Learning applied to Pong
http://www.danielslater.net/2016/05/pydatalondon-2016.html