QUESTION: Observe what you see with the agent's behavior as it takes random actions. Does the smartcab eventually make it to the destination? Are there any other interesting observations to note?

There aren't many observations to notice, except for the fact that the smartcab drives in circles.
I've run the test 5 times for 1000 trials each and got the following results:

Trials	Success	Ran Out of time	Hard Deadline
1. 1000	682	475	318
2. 1000	666	460	334
3. 1000	655	474	345
4. 1000	678	476	322
5. 1000	652	426	348

Success rate if we don't enforce the deadline (<= 100 moves) - 66.66%, if we enforce the deadline it is 20.44%.

QUESTION: What states have you identified that are appropriate for modeling the smartcab and environment?
Why do you believe each of these states to be appropriate for this problem?

I decided to include following states: light (green, red), oncoming (None, left, right, forward), left (None, left, right, forward) and a next waypoint.
We are using here the US driving laws, which means that traffic to the right don't affect our behavior on a light-regulated intersection.
Next waypoint is our goal and we need to understand can we proceed there or not.
Only the light, oncoming and left traffic can help us to understand what is going to be our next move and whether it is going to be an allowed move.

I also decided not to include the deadline as a state, because it can take so many values, that it will make a space state gigantic.

QUESTION: What changes do you notice in the agent's behavior when compared to the basic driving agent when random actions were always taken?
Why is this behavior occurring?

Smartcab started reaching destination each time while not hitting a hard-deadline.
It improves with each run and consistently achieves 80%+ success rate. It does so, because it uses a Q learning algorithm instead of taking random actions.

QUESTION: Report the different values for the parameters tuned in your basic implementation of Q-Learning.
For which set of parameters does the agent perform best? How well does the final driving agent perform?

QUESTION: 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? How would you describe an optimal policy for this problem?

Alpha is the learning rate. Setting it to 0 means that the Q-values are never updated, hence nothing is learned. Setting a high value such as 0.9 means that learning can occur quickly.

According to the lecture, it was said that it is always a great idea to decay Alpha so we can avoid local maxima/minima, which means that our car should be more confident with what it's learning, while should be less convinced by the information as it prevents the car from changing what it knows already. With the decrease of learning rate our car becomes more convinced that it's found an optimal solution.

I've also used the discount factor gamma, which models the fact future reward is worth less than immediate reward and the value of 0.1 worked out better than others.

I've tried following sets of parameters:
alphas = [1, 0.8, 0.6, 0.3, 0.1]
gammas = [0.5, 0.3, 0.1]
epsilons = [0.2, 0.1, 0.05, 0.01, 0.005]

Results are on the next 3 pages, where you can clearly see, that for 100 experiments for parameters Alpha 0.8, Gamma 0.1, Epsilon 0.005 it consistently performs with 99% success rate: 12.3 moves on average, 22.9 rewards average, 0.4 penalties average.

Q values after 100 trials with optimal parameters:
To the left: waypoint, light, oncoming traffic, left traffic, to the right – corresponding values for an action.

('forward', 'green', 'right', None): {'forward': 1.16, 'right': 1, None: 1, 'left': 1},
('forward', 'green', 'forward', None): {'forward': 1.10, 'right': 1, None: 1, 'left': 1},
('forward', 'green', None, 'forward'): {'forward': 1.22, 'right': 1, None: 1, 'left': 1},
('forward', 'green', None, 'right'): {'forward': 1.45, 'right': 1, None: 1, 'left': 1},
('forward', 'green', None, 'left'): {'forward': 2.51, 'right': 1, None: 1, 'left': 1},
('forward', 'green', None, None): {'forward': 5.08, 'right': 0.73, None: 0.73, 'left': 1},
('forward', 'green', 'left', None): {'forward': 2.00, 'right': 1, None: 1, 'left': 1},

('forward', 'red', None, 'left'): {'forward': 0.74, 'right': 0.77, None: 1, 'left': 1},
('forward', 'red', None, 'forward'): {'forward': 0.74, 'right': 0.64, None: 1, 'left': 1},
('forward', 'red', None, None): {'forward':-0.93, 'right':-0.01, None: 1.75e-67, 'left': -0.32},
('forward', 'red', 'forward', None): {'forward': 0.22, 'right': 0.31, None: 0.41, 'left': 0.21},
('forward', 'red', None, 'right'): {'forward': 0.67, 'right': 0.75, None: 0.63, 'left': 1},
('forward', 'red', 'left', None): {'forward': 0.70, 'right': 0.81, None: 1, 'left': 1},
('forward', 'red', 'right', None): {'forward': 0.37, 'right': 0.70, None: 0.83, 'left': 0.64},

('right', 'green', None, None): {'forward': 0.68, 'right': 2.29, None: 1, 'left': 1},
('right', 'green', 'right', None): {'forward': 0.76, 'right': 1, None: 1, 'left': 1},
('right', 'green', None, 'left'): {'forward': 0.47, 'right': 1.19, None: 1, 'left': 1},
('right', 'red', None, None): {'forward': 0.76, 'right': 2.22, None: 1, 'left': 1},
('right', 'red', None, 'right'): {'forward': 0.72, 'right': 1, None: 1, 'left': 1},
('right', 'red', 'forward', None): {'forward': 0.21, 'right': 1.55, None: 1, 'left': 1},
('right', 'red', None, 'left'): {'forward': 1, 'right': 1, None: 1, 'left': 1},

('left', 'green', None, 'left'): {'forward': 0.84, 'right': 1, None: 1, 'left': 1},
('left', 'green', 'right', None): {'forward': 0.68, 'right': 1, None: 1, 'left': 1},
('left', 'green', None, None): {'forward': 0.42, 'right': 0.81, None: 0.70, 'left': 3.09},
('left', 'red', None, None): {'forward':-0.19, 'right':-0.05, None: 5.47e-26, 'left': -0.05},
('left', 'red', 'forward', None): {'forward':-0.09, 'right': 0.47, None: 1, 'left': 1}

First of all, looking at the value we clearly see that we haven't explored all action/value pairs, so 100 trials are not enough for convergence.
Looking at the current values I also think I can say that this policy is optimal, as the car obeys the laws and reaches the final point in 99+% cases on time.
Penalties received are probably because of random actions, as the car didn't have knowledge for what could be the best action, for example – here we need to turn left, but car will make a random action:
('left', 'green', None, 'left'): {'forward': 0.84, 'right': 1, None: 1, 'left': 1},

Parameters	Success Rate	Steps	Rewards	Penalties
Alpha 1 Gamma 0.5 Epsilon 0.2	80%	16.7	25.9	4.4
Epsilon 0.1	89%	14.3	25.9	2.8
Epsilon 0.05	93%	13.5	27.6	2.8
Epsilon 0.01	93%	14.0	25.3	0.9
Epsilon 0.005	99%	13.2	24.3	0.6
Alpha 0.8 Epsilon 0.2	87%	17.2	23.7	3.1
Epsilon 0.1	98%	15.1	23.8	2.0
Epsilon 0.05	98%	13.1	22.2	0.9
Epsilon 0.01	99%	13.1	22.5	1.0
Epsilon 0.005	99%	12.6	23.2	1.2
Alpha 0.6 Epsilon 0.2	86%	14.7	21.3	3.0
Epsilon 0.1	96%	16.1	21.3	1.7
Epsilon 0.05	100.00%	13.6	22.0	1.1
Epsilon 0.01	98%	12.2	22.3	0.4
Epsilon 0.005	100.00%	13.6	21.6	0.6
Alpha 0.3 Epsilon 0.2	74%	16.8	24.3	4.6
Epsilon 0.1	92%	12.7	25.1	4.0
Epsilon 0.05	91%	14.6	28.1	3.9
Epsilon 0.01	99%	13.4	25.3	1.1
Epsilon 0.005	98%	14.2	24.4	1.3
Alpha 0.1 Epsilon 0.2	88%	16.5	24.4	2.8
Epsilon 0.1	98%	15.6	23.4	1.8
Epsilon 0.05	95%	14.3	22.3	1.0
Epsilon 0.01	100.00%	15.0	25.1	0.6
Epsilon 0.005	100.00%	14.1	22.9	0.6
Alpha 1 Gamma 0.3 Epsilon 0.2	88%	15.6	23.2	3.1
Epsilon 0.1	96%	14.3	23.6	1.6
Epsilon 0.05	98%	14.4	22.4	1.2
Epsilon 0.01	98%	15.7	21.3	0.5
Epsilon 0.005	100.00%	13.0	22.8	0.4
Alpha 0.8 Epsilon 0.2	78%	15.5	23.9	5.7
Epsilon 0.1	89%	15.5	26.4	4.4
Epsilon 0.05	96%	13.6	25.7	.7
Epsilon 0.01	98%	13.8	22.7	0.9
Epsilon 0.005	100.00%	12.7	24.2	3.1
Alpha 0.6 Epsilon 0.2	84%	15.3	23.3	3.2

Epsilon 0.1	95%	14.3	23.5	1.9
Epsilon 0.05	99%	14.6	23.3	1.1
Epsilon 0.01	98%	12.9	23.7	0.5
Epsilon 0.005	99%	13.1	22.4	0.6
Alpha 0.3 Epsilon 0.2	89%	15.4	21.4	3.0
Epsilon 0.1	98%	15.2	22.3	1.6
Epsilon 0.05	97%	13.4	22.8	0.9
Epsilon 0.01	99%	13.6	22.2	0.4
Epsilon 0.005	99%	13.6	22.2	0.3
Alpha 0.1 Epsilon 0.2	71%	17.5	24.3	4.4
Epsilon 0.1	77%	13.8	25.5	4.1
Epsilon 0.05	97%	14.3	25.3	4.0
Epsilon 0.01	99%	12.6	23.6	0.7
Epsilon 0.005	92%	13.8	22.5	3.6
Alpha 1 Gamma 0.1 Epsilon 0.2	92%	17.0	23.6	3.5
Epsilon 0.1	95%	14.4	24.7	2.0
Epsilon 0.05	98%	15.1	21.7	1.4
Epsilon 0.01	98%	12.6	22.8	0.8
Epsilon 0.005	100.00%	13.0	22.7	0.5
Alpha 0.8 Epsilon 0.2	89%	15.6	22.6	3.0
Epsilon 0.1	93%	13.8	23.1	1.4
Epsilon 0.05	99%	13.3	22.5	0.7
Epsilon 0.01	98%	13.1	21.4	0.4
Epsilon 0.005	99%	12.3	22.9	0.4
Alpha 0.6 Epsilon 0.2	78%	14.0	22.9	4.1
Epsilon 0.1	97%	14.9	23.7	2.9
Epsilon 0.05	95%	15.3	24.8	1.1
Epsilon 0.01	99%	12.1	22.7	0.5
Epsilon 0.005	99%	14.6	22.8	0.8
Alpha 0.3 Epsilon 0.2	87%	15.4	24.0	3.6
Epsilon 0.1	96%	14.0	22.7	2.0
Epsilon 0.05	99%	13.8	23.3	0.8
Epsilon 0.01	100.00%	14.3	22.1	0.7
Epsilon 0.005	100.00%	13.2	23.0	0.5
Alpha 0.1 Epsilon 0.2	94%	15.9	22.4	2.7
Epsilon 0.1	98%	14.8	22.7	1.4
Epsilon 0.05	100.00%	13.8	22.5	0.9
Epsilon 0.01	100.00%	14.0	22.9	0.4
Epsilon 0.005	99%	13.2	22.7	0.4