pattern recognition

Construction zones. Flaggers, non-standard traffic control, unclear lane markings.

Poor lane markings. As Steven Shladlover put it, if you can't see it, the SDC can't see it either.

Object identification. It can't always tell what road debris is made of, and this impacts how it should respond. You don't want to slam on your brakes to avoid an empty cardboard box.

Crash avoidance. Right now it seems to be brake-only, and they admit that this strategy doesn't guarantee safety. Levandowski gave an example of the car in front of you swerving to get around a couch in the road. The SDC wouldn't have seen it, doesn't have to to stop, and isn't programmed to change lanes.

Crash optimization. There also doesn't seem to be much of a plan for what to do when a crash is unavoidable. The "best" way to crash should be somewhat obvious using crash models, but it can also be a morally difficult in application. For example, a crash model could predict that colliding with an SUV is safer than with a Prius. That seems like targeting, which is unfair to SUV drivers.

Liability. Tons of threads on here about this, and a legitimate concern for automakers. Toyota paid out $1.6 billion for unintended acceleration because they couldn't prove that their software didn't malfunction. It will be just as hard to prove that SDC software is perfectly safe.