Challenges

- Requirements: specification in advance → into languages to be able to verify
- Arbitrary design methodologies and architectures → tools for subsets of systems
- Tools for all kinds of systems (not necessarily the same tool) → to compare against each other
- Simulations and models for evaluation → efficient and cost-effective
  - How do we model and consider environments with people, the real world, space, houses, animals?
  - Modelling what systems that learn/adapt online are going to do
- Extracting a model of decision making of the system (high-level) → to determine if the robot intending to do harm or not
- Cases and simulations against which the systems should be verified
- Systems that self-determine and self-assess for self-management → for subsystem verification and visibility of behaviours and faults
- Integration → how do we put together verification for control systems, decision making, hardware and software together, since autonomous systems have all of these elements together?
- Systems with emergent behaviours

Discussion

How does the self-assessment fit into traditional verification? (how will verification allow learning) A function manager administrates functionality and learning at runtime. The function manager would do the verification when the system learns. Also, at a high level, one should verify the manager.

How do you ensure that the function manager is less complex than that which you are verifying? Agent sits on top, decisions of these agents are verified.

What about a simple system? You add an agent on top. System should expose and understand what it's doing → self-determination.

Regulators care about system as a black box → defined and verified system behavior. Reasoning about components.

How can you maintain and black box the verification in complex systems and environments? This does not work for adaptive systems, otherwise you would need to test for a variety of scenarios.

Verifiability of systems needs to be evaluated. Most systems are not verifiable at all; try to build verifiable systems but there are interesting systems that are not verifiable. There are some domains that we may see more clearly, others less so.

Deep learning is impacting verification; white box verification is problematic. What are the research opportunities? How can we verify and validate systems and build trust in systems that cannot be verified by traditional methods (formal, testing). Deep learning not just perceptual, but also into control and decision making.

How do we guarantee learning systems learn good things? Life-long verification? Unlearning something may be as complex as learning. Self-assessment should also reason about what it's learning, what it should be learning and what it should not. Deep learning needs to be managed.

How do you build a system capable of understanding what it is learning? Functions as agents and reasoning about functions.

Verification of something like alpha-Go (that learns) might be limited to having oracle knowledge of what is good.

Can we verify very intelligent and complex systems? Defining what a system like that does is a problem. Let system learn, but in a contained environment. Represent safety limits as part of constraint set.

Doesn't having agent based function management doesn't just move the verification problem to the agent? If reasoning is more explicit, via the agent, it might be easier to verify, as the function management.

We might not be able to provide 100% guarantees for all the environments, for all the cases, we might have to limit the verification.

Would you agree that figuring out how to generalize a systems performance from an example or set of examples be a research challenge?

Formal models from scenarios → extendible, abstractions can be made, they help verification. Big research challenge in this area is finding the right abstractions for verification, which enable formal verification for highly complex systems. How can we better leverage existing methods to address verification problems in autonomy (to model and verify systems that learn)? Even for simple, non-learning systems, we cannot find abstractions for techniques like slicing.

Runtime adaptive systems → how to reason about them and make models? Performance is different according to the circumstances. There are runtime monitors. Function managers can help to constraint what the system does.

How do we set the boundary? How do we define what the system should and should not do? We are horrible in robotics at realizing what matters (e.g. florescent lighting.)

Instead of coming from a formal methods perspective, human-like learning verification. E.g. schooling, exams for the system that inspect system's understand of subject matter (from a formal model), certification for driving, etc.

What about cognitively complex systems? Even sometimes evaluating a student's responses on an exam may be difficult to handle quantitatively.

Rational agent is verified in a provable way, abstracts perception. Requires fully specifying responses to all possible decisions. The recognition, classification, sensor input, control systems are not considered, only the inputs they provide to the agents.

A remaining challenge → qualification of sensor systems (video, classification of environment components). Verifying rational agents involves assumptions on these other peripheral systems. They need to be verified but it’s hard. People can come up with safety constraints for systems like AUVs, etc. Iterative approach will yield increasing capable systems by increasing the constraints and system properties.

Many dependencies go into simple constraints like what is a safe distance (e.g. uncertainties in data, etc).

Licensing model of verification: weakness is that such tests make assumptions on key aspects like assuming the driver is not a psychopath… so those aspects aren’t tested traditionally. Problem of requirements and specifications. Processes are based on assumptions about what humans already know; robots don’t have them. Fundamental problem is intractable.

Baseline could be system makes decisions based on human decisions, but humans make mistakes all the time – how does verification work in this way? Rational specification of high-level properties of ethical/abstract choices to avoid prescribing all possible instances.

One of the gaps left to discuss – how do we define of when to stop enumerating cases? (e.g. child vs. dog, crawling child vs. dog, etc…)