All reviewers - Comparison with Ellis 2018. A crucial component of Ellis 2018 is the ability to incrementally render specifications inferred from the input image and measure similarity with that image. This similarity is used as a Bayesian likelihood for specifications sampling. However, this is not applicable to our setting as there is no direct equivalent to the incremental rendering process. Our approach offers more generality for neuro-symbolic systems.

Action/perception primitives. In Section 2.2, we opted for an abstract definition of action and perception primitives as these are domain dependent. For instance, Vizdoom actions include "move forward", "turn left" and "attack". Vizdoom perceptions include "is there a monster?" or "is a revenant in target?". We will add a detailed list in the paper.

Reviewer 1 - Discarding single tokens instead of entire traces. This is a promising direction for future work to improve sample efficiency. Yet there is one fundamental difference with Barman 2010. Angelic programming allows for uncertainty in the program, while the approach proposed by the reviewer introduces uncertainty in the specifications. We agree that there is a conceptual connection, but it is not obvious to us whether both problems are equivalent.

Comparison to Chen 2019: Execution-guided neural program synthesis. We do not compare because the synthesis tasks are different. To cite Chen 2019 (about demo2program) "Sun 2018 propose to synthesize the program from demonstration videos, which can be viewed as sequences of states. In such a problem, all intermediate states can be extracted from the videos. On the contrary, in the input-output program synthesis problem studied in our work, the input to the program synthesizer provides only the initial state and the final state." Chen 2019 infers likely intermediate states in order to simplify synthesis, while in our case they are part of the specification. This makes the tasks incomparable.

Related work. We will refer to Solar-Lezama 2008 for the concept of program sketching, and improve the dichotomy between "rule-based" and "statistical" methods. We will correct the reference to DeepCoder.

Progressively decrease cost bound until problem is unsat. We have implemented an alternative decremental scheme, preliminary results do not show a noticeable performance gap. We thank the reviewer for the literature references.

Accuracy (%)	Karel			ViZDoom		
PLANS (best configuration)	91.6	53.9	34.2	88.0	65.5	58.8
Decrement cost bound (best configuration)	92.1	53.8	34.5	87.5	64.8	57.8

Reviewer 2 - Rigidity of the approach. The filtering technique developed to deal with missclassified perceptions primitives does not make domain specific assumptions and is potentially applicable to similar neuro-symbolic systems.

Cost of iterative search over multiple attributes. A possible solution to this problem is to directly encode the cost function over programs in the solver. This is not possible in Rosette, that we chose for its flexibility in encoding DSLs. Future work could attempt to build on different solvers integrating cost functions and compare performance.

Impact to the NeurIPS community. We believe that developing efficient neuro-symbolic systems is crucial to the machine learning community. Dealing with specification uncertainty is one of the major challenges involved.

Softmax scores vs. ensemble methods. Our method is orthogonal to the choice of uncertainty measure. Ensemble techniques provide a good measure of uncertainty, but there is a high computational cost involved with training sufficiently many members. This is why we followed prior work using softmax response, which yields good performance.

Reviewer 3 - No thresholds for Karel. There is no need for filtering as the network reliably infers the specifications.

Perception primitives. No semantic understanding of the perception primitives is required for our method. Besides, our filtering technique introduces some resistance to noisy training labels as a side effect. Concerning scalability, our empirical observation is that program length and control-flow depth matter more than number of primitives.

Encoding of actions/perceptions into I/O specifications. High-level semantics of the encoding are indeed intuitive. The technical challenge lies in implementing these semantics accurately in the Rosette solver based on Racket.

Technical contributions. The ability to deal with uncertainty in specifications has been repeatedly described as a crucial aspect of neuro-symbolic systems, since traditional symbolic solvers are extremely vulnerable to noise. We addressed the key technical challenge of dealing with misspecifications in a generic way.

Generality of the action/perception framework. It is a very natural way of describing agent-environment interaction. An important direction for future work is to generalize to arbitrary combination of perceptions in conditionals.

Advantages of removing program supervision. (i) action/perception labels are much easier to acquire. (ii) it removes the need to train on several different demonstrations of the same program . (iii) it drastically reduces training cost.

Reviewer 4 - Experimental comparison with Raychev 2016. This is problematic, because the algorithm in Raychev 2016 deeply modifies the traditional program synthesis process. It was only evaluated on bit-stream programs, and has not been integrated in a general purpose solver yet. In contrast, our filtering algorithms build on top of state-of-the-art solvers, and are simple to implement. We have contacted the authors of Raychev 2016 and agree on additional advantages of our approach in a neuro-symbolic setting (i) it focuses on satisfying the most confident samples, which makes finding the correct program more likely. (ii) sorting the samples by confidence reduces overall time complexity.

Program size - scalabilty. Programs are composed of up to 43 tokens, and we observe balanced time measurements between neural network inference and symbolic solver calls. We believe that parallelism inside solver calls could help scaling to larger programs with more control-flow statements. This would be fair as the network runs on a GPU.