Dear editor, dear reviewers,

We would like to take this chance to thank you all very much for the valuable feedback and appreciate that you found the time to evaluate our manuscript. We took your advices at heart to improve, refine, and polish our work and the manuscript. With this reply we like to address your comments and point-out how we have reflected them in a refined version which we hope will find your approval now.

Response to Reviewer 1

In consultation with the editor, all changes to our manuscript in response to the reviews are highlighted with a red color directly in the manuscript.

Point 1:

This reviewer spotted only a handful of typos, which are listed at the bottom of this review.

- p.6 line 176: word "fusion" repeated
- p.9 line 304 method -> methods
- p.13 line 434 & 437: Cursor -> Cursors
- p.17 line 582: to integrated -> to integrate

Response: We corrected the typos found by reviewer 1. Additionally, we checked the manuscript for spelling mistakes and grammar.

Point 2:

The state of the art is complete enough, with maybe the omission of symbolic-statistical fusion approaches such as e.g.


Response: We included the suggested references in our manuscript to further complete our related work. We introduce symbolic-statistical fusion methods in section 1. *Introduction* fourth paragraph. Three paragraphs later we bring them into the context of our work.

**Point 3:**

One question on that topic: can context be dynamic? E.g. can the approach presented by the authors handle information such as time, position of the user, or environmental information such as the surrounding light level?

Response: To answer the reviewers question: Yes it is possible to access dynamic context like time, position of the user, or environmental information as long as it is represented in the application state. We use semantic queries to directly access the application state and behavior of a RIS in real-time during live simulation. We reformulated our manuscript in the following paragraphs to better highlight this feature:

- In section 2.2.2. *Context* second last paragraph
- In section 3.2.5. *Design Decision: Semantics-based State- and Behavior Management Techniques* first and second last paragraph

**Point 4:**

On page 3, lines 89-90, the authors mention that "there is currently no solution on how to fulfill two fundamental semantic fusion requirements (see Table 1): handling probabilistic and chronologically unsorted input". This is true for finite state automatas-based approaches, but approaches such as meaning frames or unification-based approaches have been able to handle such inputs from the start (see e.g. ref. [30] from the article)... However with the limit that they have to "wait" for all input to be processed before integration can start. There is a little interesting discussion here on which the authors could build.

Response: Reviewer 1 is absolutely right. Unification as described in [30] is capable of handling probabilistic and chronologically unsorted input. However one of its weaknesses, as the reviewer mentions, is that this approach has to "wait" for all inputs before semantic fusion can occur and thus is not able to provide continuous feedback, which is especially important when dealing with probabilistic multimodal input. We should have pointed this out more precisely, hence, here we now elaborate on this issue at the start of section 4. *Discussion*.

**Point 5:**

Authors mention that they plan to assess how well their approach works performance-wise. This is actually a point that this reviewer would have liked to see developed a bit further. In the current experience of the authors, how well does their approach cope with high frequency data sources, such as e.g. a touch surface...
such as the one pictured in Figure 1 middle? If both users place their full hands on the surface at the same time, what happens?

**Response:** At the end of section 4. **Discussion** we added a paragraph which details limitations of the cursor concept and our overall system, as well as, elaborates on performance optimizations we already apply in several of our demonstrations to handle high frequency data sources. Additionally, we designed and conducted a comparative performance benchmark. We compare our method with a state of the art ATN approach in section 3.4. **Benchmark.** The results validate the fulfillment of the principle performance characteristics which are needed for highly interactive systems. The benchmarks also highlights a beneficial run-time behavior of the proposed concurrent cursor implementation when it comes to handling an increasing number of hypotheses.

**Point 6:**

The main weakness of the article revolves around the lack of validation of the approach. Evaluation is mentioned in the future works at the end of the article, however, mostly on performance aspects. Beyond performance aspects, a core question is the usability (understandability, expressivity) of the cATN approach and language w.r.t developers. The authors would be encouraged to explore this side of their work in further publications. This would also build on the (very good) argument the authors make in page 2, line 75, about typical interaction design techniques. Also on the topic of understandability and expressivity of the proposed language, the fact that it relies on Scala is definitely a positive aspect (avoiding the need for developers to learn a brand new language). However, it seems to this reviewer that the proposed approach would be well suited for the creation of a visual tool for easier definition of cATNs. These last points are more general suggestions and don't have to be addressed in a revision of this article.

**Response:** The reviewer is absolutely right and in similar work we did investigate into so-called API usability questions. Besides the now included performance comparisons, API usability is a very important aspect and crucial for facilitating rapid prototyping of multimodal interfaces. At the end of section 4. **Discussion** we further elaborated on API usability and problems with regard to measuring and comparing API usability. Testing API usability is kind of a tricky beast since it not only needs a sufficient number of participants but these participants also need to a) work with the API for a sufficient long time and b) must represent a representative sample from novice developers to professionals, ideally all with good knowledge of multimodal interface development. We try to accomplish this by integrating the method into our classes on multimodal interfaces but did not collect enough data right now since it takes several semesters to collect a good sample. We did monitor qualitative feedback though to optimize the API during the development. As pointed-out by the reviewer we are planning to pursue on this topic in future work.