We would like to thank the reviewers for insightful comments which help improving the clarity.

We will include a discussion comparing AVTA with other convex hull vertices detection algorithms.

We will also include a detailed description on the binary search type method in case \gamma is unknown. Also, more information and discussion about experiment will also be included in the supplement.

We would also like to stress that the application of vertices based method discussed in the paper requires robustness property of algorithm in convex hull problem. Our work provides the first robust guarantees for vertices detection for general convex hull. Below we provide answers to specific questions:

- How does AVTA compare in runtime to other algorithms for finding the convex hull? In particular, a standard algorithm in higher dimensions is to my understanding QuickHull, which has average complexity O(n log n). This is comparable to the (roughly...) O(n K) of AVTA for the common case of K ~= log(n). The gift wrapping algorithm I believe has complexity O(n K).

- QuickHull/giftwrapping are exact, and don't have some of the complexity surrounding the approximations arising from use of the triangle algorithm. What are the benefits of AVTA here?

Chazelle et al [10] showed that for general d dimension, the optimal complexity to compute exact convex hull is O( nlogn + n^(d/2)). The average complexity O(nlogn) for QuickHull is for low dimensions (d=2,3). (Sartipizadeh 2016.)

Computing an approximation of vertices can be much cheaper as discussed in the paper.

In addition, exact algorithms can output much more vertices than desired. In practice, one only need a small subset of vertices with ‘high quality’ among all vertices. The AVTA algorithm find vertices progressively thus achieves both efficiency and robustness for practical applications.

- Similarly, for the robustness properties: is this AVTA specific? The pruning seems like it dovetails nicely with the tolerances that can be set for the triangle algorithm, but requires some prior knowledge as to \sigma. I do like the random projections approach, but that seems somewhat unrelated to AVTA.

In practice where only the number of vertices desired is given and there is no prior knowledge on the parameter, the AVTA will output a super set of vertices. The super set of vertices is pruned in a non-parametric way. (i.e. Greedy algorithm). This part is not discussed in detail in the paper as our analysis is based on some prior knowledge of \sigma.

- Is there further explanation of the insight for using random projections? Specifically, "If the perturbed set is randomly projected onto a lower dimensional space, it is more likely for an original vertex to still be a vertex than for a spurious vertex." I couldn't find any proof or discussion of this claim in the supplementary material.

This is an empirical observation. Investigating the theoretical support of this phenomenon is an open problem for future work.

- The AVTA+CatchWord idea seems quite interesting, but it's not clear to me to what degree AVTA and the triangle algorithm are required. Would any convex hull algorithm (+ standard linear programming to find coefficients w.r.t. vertices) also work to embed the documents?

Computing vertices by standard linear programming is inefficient and there is no guarantee for the robustness. (Arora et,al. 2012 [1]) To embed the documents, one needs to find a small set of vertices of convex hull which can well represent each points. Here, AVTA is used for this purpose.

In short, there are several interesting claims/insights presented, but it's a bit difficult to disentangle them and determine how they relate to/depend on the central AVTA technique (vs. other ways of obtaining vertices of the convex hull). It seems like much of the paper doesn't depend on AVTA specifically, and it's not obvious to me that AVTA is a better way of finding convex hulls compared to existing approaches.