KNN has several drawbacks for predicting real time location. The model is not easily transferable without the training data, its speed decreases with more observations, and it could have difficulty with outliers.

KNN predictions are made by comparing a point to its closest neighbors then determining its location by either voting or an average of its neighbors’ locations. This means you must have other data points to make a prediction. If the training set is large, it would need to be copied to every location where the model would need to be run. If it is running on a single server somewhere that might not be an issue, but if it is running on user’s cell phones that might make the transfer time consuming and the storage cost to the user might be prohibitive.

Large training set sizes could also be an issue when it comes time to compute the prediction. The more observations in the training set, the more need to be searched through to find the nearest neighbors, increasing prediction time. The cost of this would be dependent on the use case. If it is merely observational, perhaps a few seconds difference might not matter. But it a decision needs to be made quickly based on location, then it could be a significant problem. For instance, if a robotic currier needed to decide where to turn based on its location, or trying to intercept an item moving through a storage facility.

Finally, there could be issues with outliers or irregular spaces. A new data point far outside the space would be placed by the average of its three closest neighbors, which could be far away from its actual location. I think that would manifest itself as a “ghost”, where the system would say something was present when the signal was coming from outside the space. Depending on the sensors used, there might be some maximum distance the sensors can transmit which might mitigate this issue. Another issue might be when the nearest neighbors might place an object outside the space.

A regression or random forest approach might solve the first two issues. These models are more portable in that they do not require the training data to make predictions and faster because they don’t need to search the training set for nearest neighbors. To address the outlier issue, we could limit our predictions based on the boundaries of the space. Any predictions outside the space are move to the nearest in-space location. We could potentially deal with outliers in preprocessing, perhaps looking for signal strengths that don’t make sense for the space.