# Logboek warsimulator project

* Sorteeralgoritme van de health bars aangepast, insertion sort vervangen met count sort sorteer algoritme.

Oude big O van insertion sort was O(N^2) dus exponentieel, huidige count sort heeft een big O van O(n+k) waar de health (k) 100 is.

* Range check algoritme aangepast, zogenaamde “naïeve methode” vervangen door spatial hash.

Korte uitleg: In de huidige simulatie controleerde iedere tank ALLE andere tanks in de gehele simulatie herhaaldelijk op collisie. Met de spatial hash controleert hij alleen de tanks die zich in dezelfde cel als zichzelf waardoor er minder checkt plaatsvinden en dus een duidelijke versnelling zichtbaar is.

* Toegevoegd dat de tank in een 3x3 cel radius om zijn huidige cel controleert om ook de collisies die op de borders van de cellen plaatsvinden meegenomen worden.

Oude big O notatie update tanks van de naïeve methode was O(N^2), nieuwe big O bij het aanmaken is nu O(N). Collission is nog steeds O(N^2) maar in de praktijk is het toch sneller.

* Spatial hash ook op raketten toegepast.
* Op verschillende plekken kopieën weggehaald en vervangen door references. Leverde de ook weer een kleine versnelling.
* Threadpool gebruiken voor multithreading op update tanks, rockets, explosions en smokes.
* Locking toegepast in tanks en rockets om exceptions op te lossen.
* Update tanks zoals bovenstaan nog steeds N^2, maar dit is alleen het geval wanneer alle tanks in 1 cel zitten. Dit kan niet doordat ze elkaar wegdrukken bij collisie.

T1 = N^2 en dan T∞ = n waar n dan voor het aantal tanks staat.

* Update rockets controleer in de 3x3 radius of er een tank in collisie range zit.

Volgens ons is de work en span hier:

T1 = n \* m en dan is T∞ = m waar n voor raketten staat en m voor tanks.

* Eerst werden alles smokes stuk voor stuk gedaan, dit is verbeterd met threading.

T1 = n en dan T∞ = 1 met dan n als aantal smokes.

* Voor explosion geldt eigenlijk hetzelfde als de smokes. Dus ook weer:

T1 = n en T∞ = 1 maar dan staat n voor explosions.