OPTIMALISEREN VAN CONTAINERPLAATSING

Opgesteld door: groep 4

Richal Rambaran	**19029217**	**19029217@student.hhs.nl**
Akram M'barek	**19066171**	**19066171@student.hhs.nl**
Jesse Huizing	**18053580**	**18053580@student.hhs.nl**
TJ Herdigein	**18140572**	**18140572@student.hhs.nl**
Ahmad Alkhatib	**18019234**	**18019234@student.hhs.nl**

1 Abstract

In dit onderzoek is onderzocht in hoeverre het plaatsingsprobleem van containers op een kade kan worden opgelost. Dit is gedaan door middel van Reinforcement Learning (RL). RL is een vorm van machine learning waarbij een agent wordt getraind om in een environment beslissingen te nemen. In het kader van het efficiënt plaatsen van containers op een kade, is de PPO-agent ingezet om te trainen door middel van interacties met een gesimuleerde container environment. Hierbij neemt de PPO-agent beslissingen over waar containers het best geplaatst moeten worden. Deze besluitvormingen zijn op basis van optimale hyperparameterwaarden, ingerichte rewards en penalties gedurende het trainen verbeterd. De resultaten laten onder andere zien dat het gefinaliseerde agent het best presteert met een learning rate van 0.00003. Bij een waarde van 0.03 presteerde de gefinaliseerde agent het slechtst. Hierbij werd een trainingstijd gehanteerd van 500.000 stappen. Door gebruik te maken van de kracht van de PPO-agent, zorgt de efficiënte plaatsing niet alleen voor het besparen van tijd en geld. Het biedt namelijk ook een basis voor vervolgonderzoek naar uitbreiding van meer werkelijke situaties.

2 Inleiding

2.1 Aanleiding

Cofano is een IT-bedrijf dat zich voornamelijk richt op softwareoplossingen binnen de logistiek. Een van de gebieden waar het bedrijf werkzaam is, is het in- en uitladen van containers op schepen, dit gebeurt op een terminal (Lewandowski, 2014). Deze containers worden verplaatst door 'reachstackers'. Dit zijn rijdende voertuigen die containers kunnen verplaatsen. In de scheepvaartindustrie wordt tijd gezien als geld. Wanneer een schip lang aan de kade ligt, omdat er gewacht moet worden op andere containers, zal dit onnodig veel geld kosten. Het is de taak van Cofano om dit proces efficiënter te maken en hiermee de kosten te minimaliseren (Cofano, sd).

2.2 Probleemstelling

Een container moet in zo min mogelijk stappen door middel van een 'reachstacker' op een schip worden gezet. Hoe meer stappen hierbij nodig zijn, des te inefficiënter het proces wordt uitgevoerd. Denk hierbij aan de afstand tussen de benodigde containers en het schip dat geladen moet worden. Zo kan een benodigde container op een onbereikbare plaats gezet wordt, zoals onderaan een stapel containers of ver achterin op de kade. Al deze extra handelingen en afgelegde afstanden kosten tijd en dus ook geld.

2.3 Doelstelling

Het doel van het onderzoek is om te bepalen waar de containers geplaatst moeten worden op de kade vanaf het moment dat de containers de haven binnenkomen. Het uiteindelijke doel is om zoveel mogelijk verspillingen in het proces te minimaliseren, zoals het afleggen van onnodige afstand en handelingen. Dit zorgt ervoor dat de desbetreffende containers op efficiënte wijze ingeladen kunnen worden op schepen door machines zoals een reachstacker.

2.4 Hoofdvraag en deelvragen

Om de hierboven genoemde doelstelling te kunnen behalen, zijn de volgende hoofd- en deelvragen geformuleerd:

Hoofdvraag

Hoe kan ervoor gezorgd worden dat containers op een gesimuleerde kade op een efficiënte manier geplaatst kunnen worden, zodat de afnemers van de containers gemakkelijk bij hun containers kunnen komen.

Deelvragen

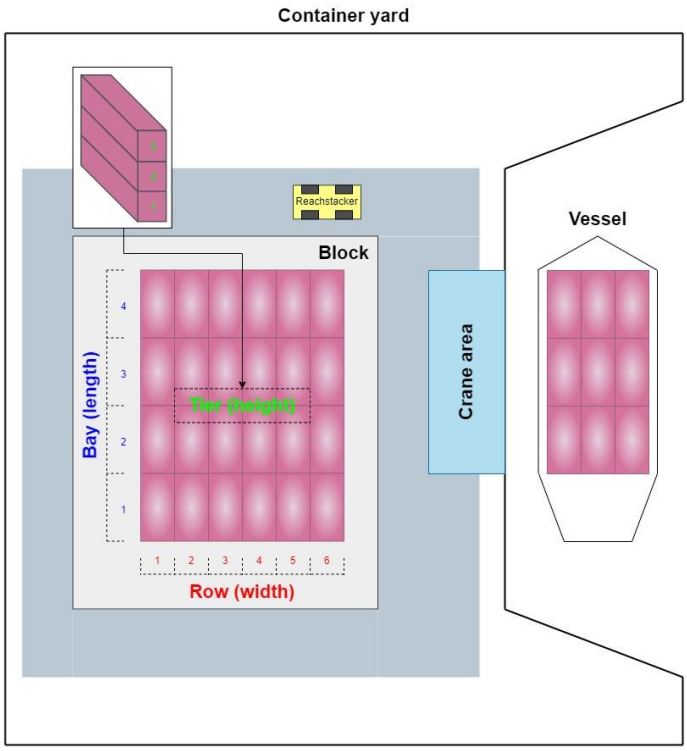
- Hoe ziet de lay-out van een kade, waar containers worden geplaatst, eruit?
- Welke RL-agents en methodes zijn relevant om dit optimalisatieprobleem aan te pakken?
- Waaruit moet de environment uit bestaan voor een goede interactie met een RL-agent?

3 Literatuuronderzoek

Om de opdracht goed te begrijpen is informatie nodig over hoe containers momenteel verwerkt worden op kades. Daarbij is ook informatie nodig over bepaalde terminologie die gebruikt wordt binnen het vakgebied.

3.1 Indeling Terminal

Figuur 1 weergeeft een plattegrond van een voorbeeld van een containerterminal. Bij deze afbeelding is er ingezoomd op één block van de terminal hierbij zijn ook de meest relevante termen ten behoeve van het onderzoek genoemd. Een block wordt gezien als een kleiner vak in de terminal waar containers (gestapeld) staan (Ntriankos, 2022). Een block bestaat vervolgens weer uit rows, bays en tiers. Dit zijn de benamingen voor de 3 dimensies waarin containers opgestapeld kunnen worden (Shih-Chan, Jaw-Shen, Shen-Long, & Flor Melina). Daarnaast rijden er op de terminal reachstackers rond. Dit zijn voertuigen die verantwoordelijk zijn voor de verplaatsing van containers over de kade als een kraan hier niet bij kan (indiamart.com, sd).



Figuur 1: voorbeeld terminal

3.2 Processen containerterminal

Het primaire proces van een containerterminal, is het vervoeren van vracht in containers tussen zee en kade. Voordat het schip aankomt wordt er vooraf een laad en los plan gemaakt tussen de kapitein van het schip en de operator van de terminal (Ntriankos, 2022). Alle verschillende containers krijgen een ID toegewezen waarna ze vervolgens worden opgeslagen in de terminal. Dit wordt gedaan aan de hand van kranen en reachstackers. Een belangrijk gedeelte van deze terminal is de gate. Hier worden de containers geregistreerd voordat ze naar binnen of buiten gaan. Tijdens dit onderzoek wordt de focus gelegd op de optimalisering van de manier waarop containers worden opgestapeld aan de hand van reachstackers.

3.3 Reinforcement Learning

Om antwoord op de hoofdvraag te geven, zal er gebruik worden gemaakt van RL. Dit is een machine learning principe dat is gebaseerd op het belonen van goede keuzes en het bestraffen van slechte keuzes (Singh, 2018). RL bestaat uit een “environment” en een “agent”. In deze environment kan de agent keuzes maken, waar een beloning aan vast zit in de vorm van punten. Bij een goede keuze zal de beloning positief zijn en bij een slechte keuze zal deze negatief zijn. Het is uiteindelijk aan de agent om in het environment de hoogst mogelijke score te behalen. Dit probleem is voornamelijk geschikt voor RL omdat er een herhalende actie is, namelijk het plaatsen van een container. Ook het feit dat men weet wat een goede en slechte plaatsing van containers is aan de hand van condities. Deze condities zijn bijvoorbeeld dat containers met dezelfde bestemming naast of op elkaar komen te staan. Dit kan goed gekoppeld worden met de agent als het aan bepaalde condities voldoet.

4 Methodologie

RL maakt gebruik van een aantal agents die getraind kunnen worden voor verschillende doeleinden. RL maakt gebruik van het simuleren van een environment waarin een agent acties kan uitvoeren. Aan de hand van deze acties ontvangt de agent rewards of penalties. Rewards voor goede acties en penalties voor slechte acties. De agent zal proberen zo veel mogelijk rewards te verzamelen.

4.1 Environment

De environment waarin de agent containers kan verplaatsen bestaat uit een blok. Een blok bestaat uit rows, bays en tiers. Voor dit onderzoek wordt er gebruik gemaakt van een blok bestaande uit 3 rows, 3 bays en 1 tier. Dit zorgt voor totaal 9 locaties waarop containers geplaatst kunnen worden. De environment bevat ook een kade waaraan schepen aangemeerd zijn. Deze schepen zijn gevuld met de containers die verplaatst zullen worden. Voor dit onderzoek wordt er gebruik gemaakt van een kade met 2 schepen met elk 4 containers. Dit zorgt voor totaal 8 verplaatsbare containers. Elk schip bevat een containergroep met een verschillende bestemming

4.2 Agent: PPO vs. A2C

Om RL toe te passen, is er een Environment en Agent nodig. In ons geval zijn PPO en A2C de meest geschikte agents voor het behalen van de resultaten die benodigd zijn. PPO staat voor Proximal Policy Optimization (Schulman, Wolski, Dhariwal, Radford, & Klimov, 2017) en A2C staat voor Advantage Actor-Critic (Mnih, et al., 2016). Volgens onderzoek is gebleken dat PPO beter werkt. Het grootste verschil tussen deze 2 agents is dat PPO een on-policy algoritme is terwijl A2C een off-policy algoritme is. A2C maakt gebruik van de voordeelfunctie om het beleid te updaten. PPO is ook een Deep Reinforcement Learning (DRL) algoritme (Bick, 2021). Het outputgedrag ziet er grotendeels uit als een standaard REINFORCE-algoritme. Dat wil zeggen dat het willekeurig bepaald wordt. PPO helpt het probleem van A2C op te lossen door te voorkomen dat het tijdens een bepaalde trainingsronde zo sterk wordt beïnvloed. Dit kan het eindresultaat behoorlijk beïnvloeden.

4.3 Rewardfunctie

De eerste versie van de rewardfunctie had alleen voor het goed plaatsen van een container een reward van +10 en voor het verkeerd plaatsen van een container een penalty van -20. De penalty was zwaarder dan de reward waardoor de agent niet goed werd getraind. In de tweede versie van de rewardfunctie werd dit probleem opgelost door de reward en penalty meer in balans te brengen. In versie 2 wordt als eerst gekeken naar beschikbare locaties. Als de container op een lege locatie wordt geplaatst, ontvangt de agent een positieve reward (+20). Probeert de agent een container op een locatie te plaatsen waar er al een container is, dan krijgt hij een negatieve reward (-20). Ook wordt er bijgehouden hoe vaak de agent een container probeert te plaatsen op een locatie waar er al een container staat. Dit wordt gezien als een illegale actie. Wanneer een container op een beschikbare locatie wordt geplaatst, wordt er naar zijn omgeving gekeken, namelijk links en rechts. Er wordt gekeken of op deze buurlocaties al containers staan. Als dat het geval is, wordt er gekeken of die buurcontainers overeenkomende bestemming hebben met de container die geplaatst is. Als dit het geval is, krijgt de agent een positieve reward (+10) per overeenkomende bestemming. Als dit niet het geval is, krijgt de agent een negatieve reward (-10) per overeenkomende bestemming. De agent eindigt met het plaatsen van containers wanneer alle containers op de kade zijn geplaatst of wanneer het aantal illegale moves gelijk is aan of groter dan 4. De prestatie werd beter met de tweede rewardfunctie, omdat de episode vroegtijdig werd afgesloten wanneer er meer dan 4 illegale moves werden gemaakt. Het eindigen vormt 1 afgeronde episode.

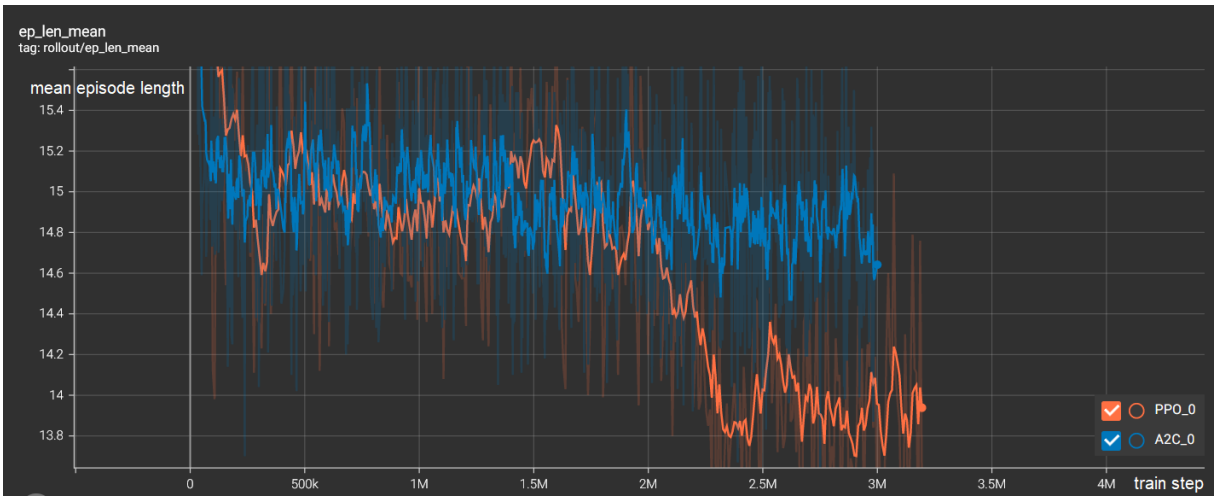
4.4 Hyperparameter tuning

De agent dient optimale waarden te hebben om zijn performance te optimaliseren. De performance van een agent is namelijk grotendeels afhankelijk van hyperparameterwaarden, naast de rewardfunctie. Hiervoor zal de agent meerdere keren getraind worden met verschillende waarden voor verschillende hyperparameters. Hieruit kan afgeleid worden welke hyperparameterwaarden zorgen voor optimale performance.

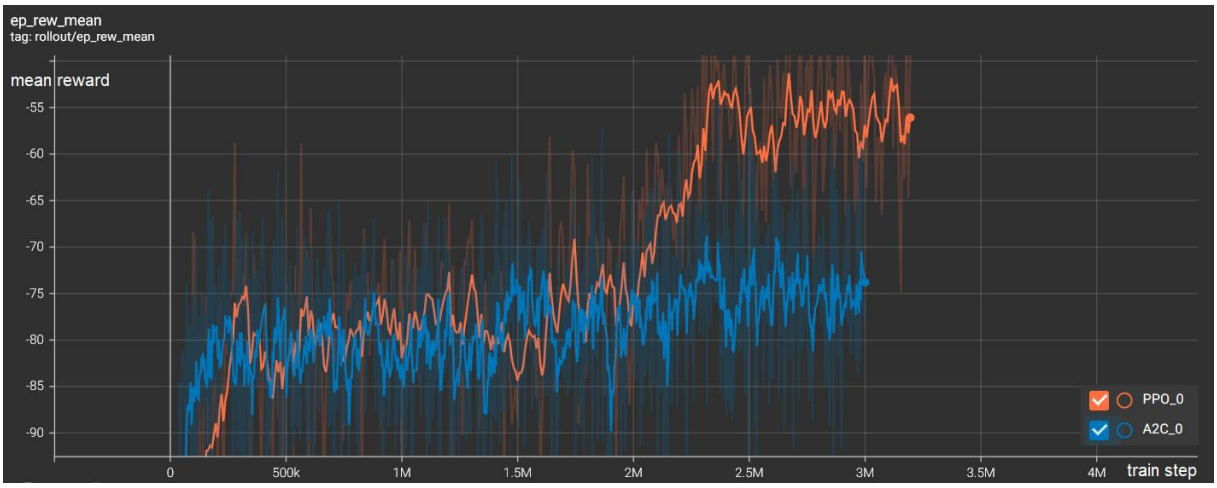
5 Resultaten

5.1 Agent: PPO vs. A2C

Hieronder volgen de resultaten van de twee gebruikte algoritmes. Het verschil in performance is duidelijk te zien. Figuur 2 weergeeft de gemiddelde episodelengte per trainingsstap aan en figuur 3 weergeeft de gemiddelde reward per trainingsstap. Er is te zien dat PPO een kortere gemiddelde episodelengte heeft en een hogere gemiddelde episodereward vergeleken met A2C.



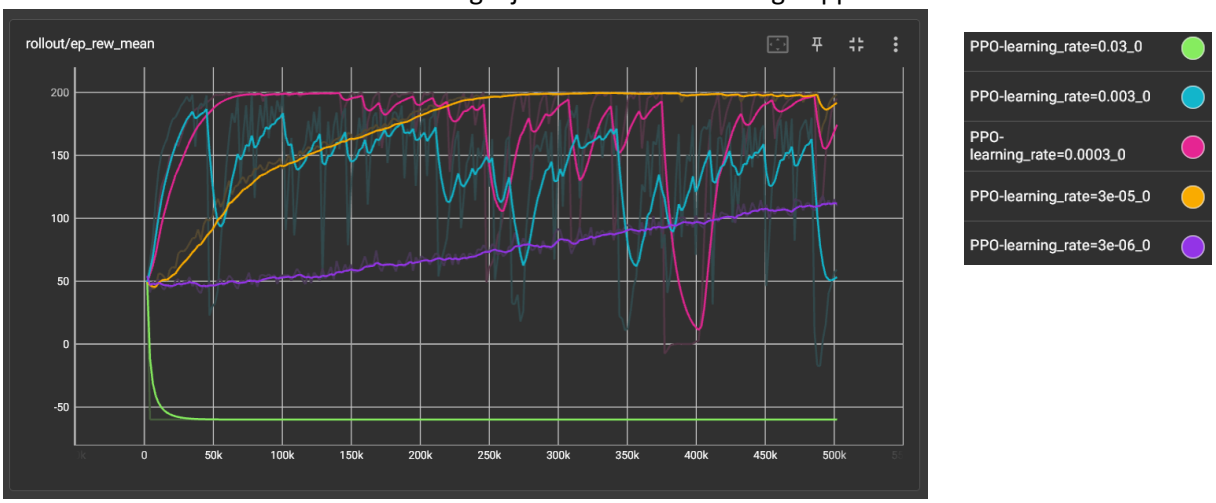
Figuur 2: de gemiddelde episodelengte per trainingsstap



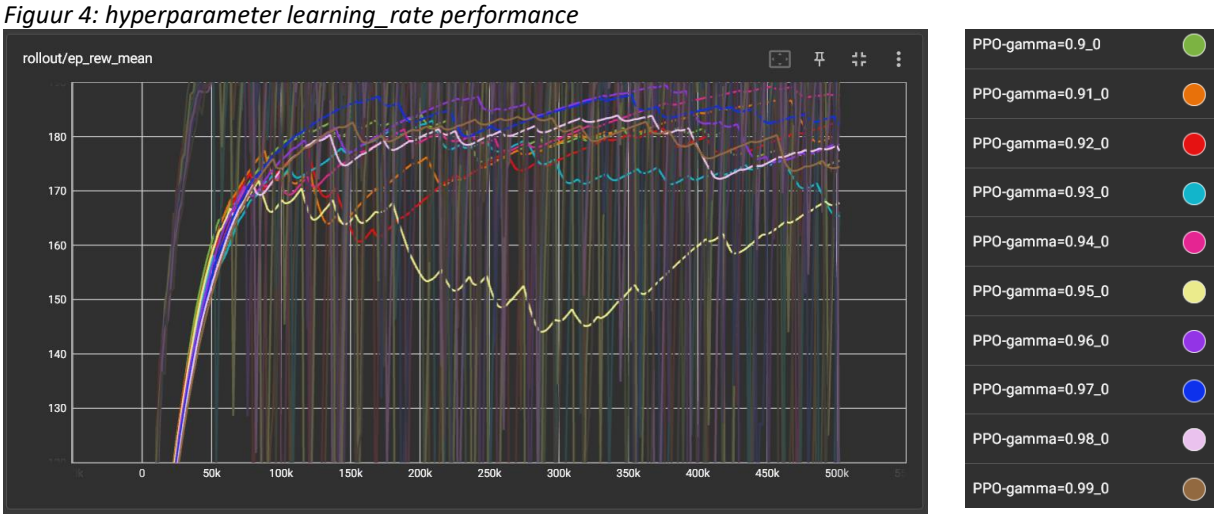
Figuur 3: de gemiddelde episodereward per trainingsstap

5.2 Hyperparameter tuning

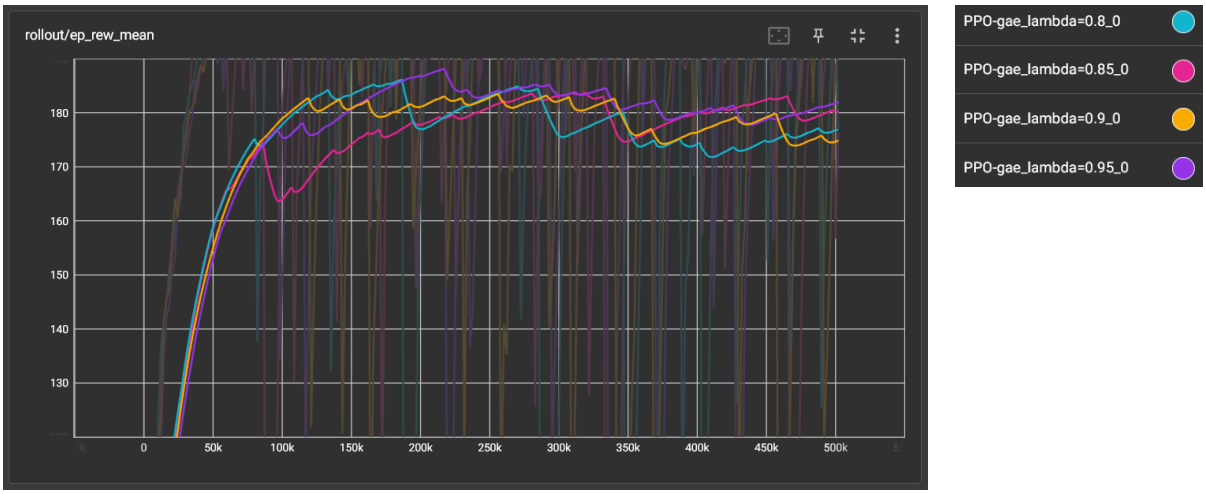
De PPO-agent is voor 5 hyperparameters getuned. Figuur 4, 5, 6, 7, en 8 weergeven de hyperparameters learning_rate, gamma, gae_lambda, ent_coef en vf_coef en hoe ze presteren met verschillende waarden over een trainingstijd van 500.000 trainingstappen.



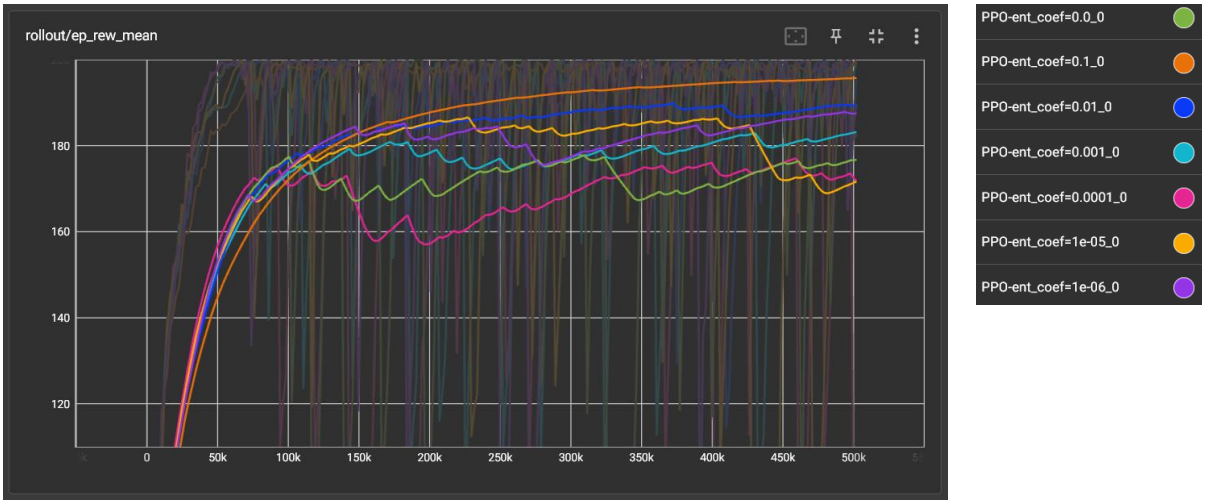
Figuur 4: hyperparameter learning_rate performance



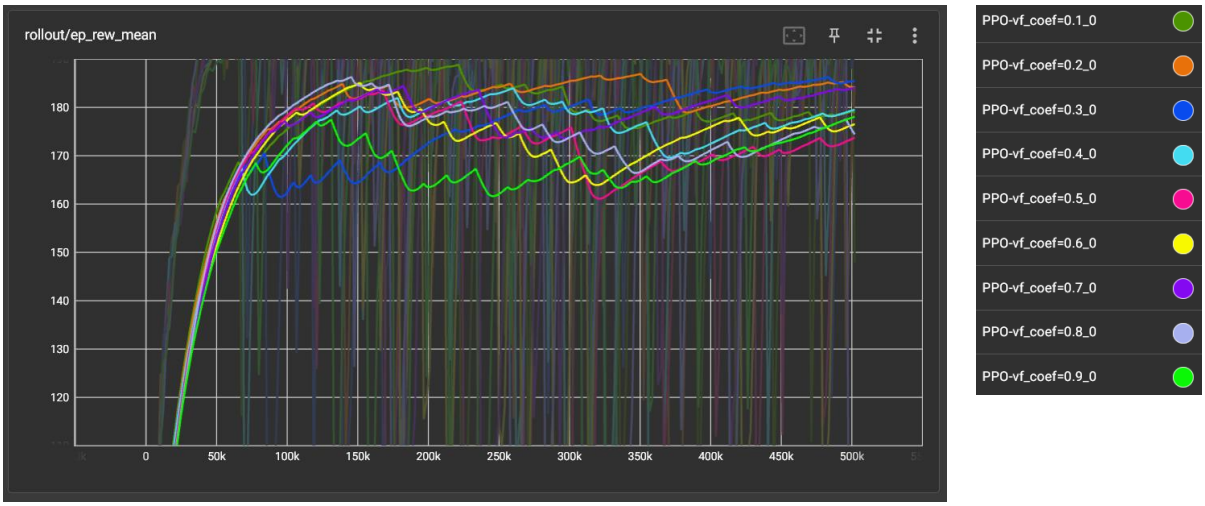
Figuur 5: hyperparameter gamma performance



Figuur 6: hyperparameter gae_lambda performance



Figuur 7: hyperparameter ent_coef performance



Figuur 8: hyperparameter vf_coef performance

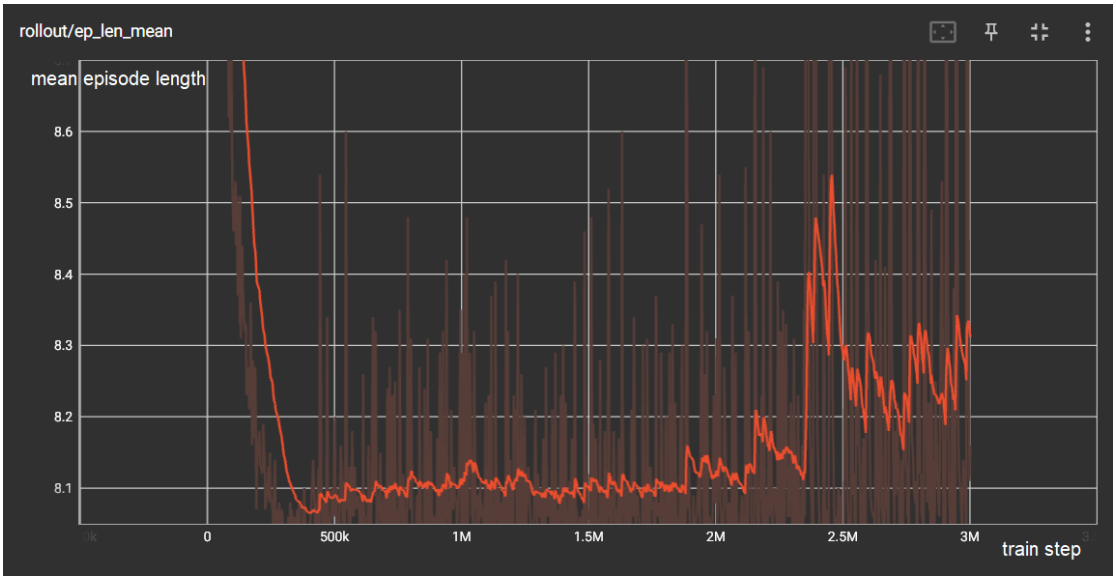
Figuur 9 geeft weer welke waarden zijn gebruikt voor de desbetreffende hyperparameter en welke waarden het beste presteren met een groene kleur.

Hyperparameter	Waarde 1	Waarde 2	Waarde 3	Waarde 4	Waarde 5	Waarde 6	Waarde 7	Waarde 8	Waarde 9	Waarde 10
learning_rate	0.03	0.003	0.0003	0.00003	0.000003					
gamma	0.90	0.91	0.92	0.93	0.94	0.95	0.96	0.97	0.98	0.99
gae_lambda	0.80	0.85	0.90	0.95	n.v.t.					
ent_coef	0.0	0.1	0.001	0.0001	0.00001	0.000001	0.0000001			
vf_coef	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	

Figuur 9: hyperparameter waarden met de beste waarden groen gekleurd

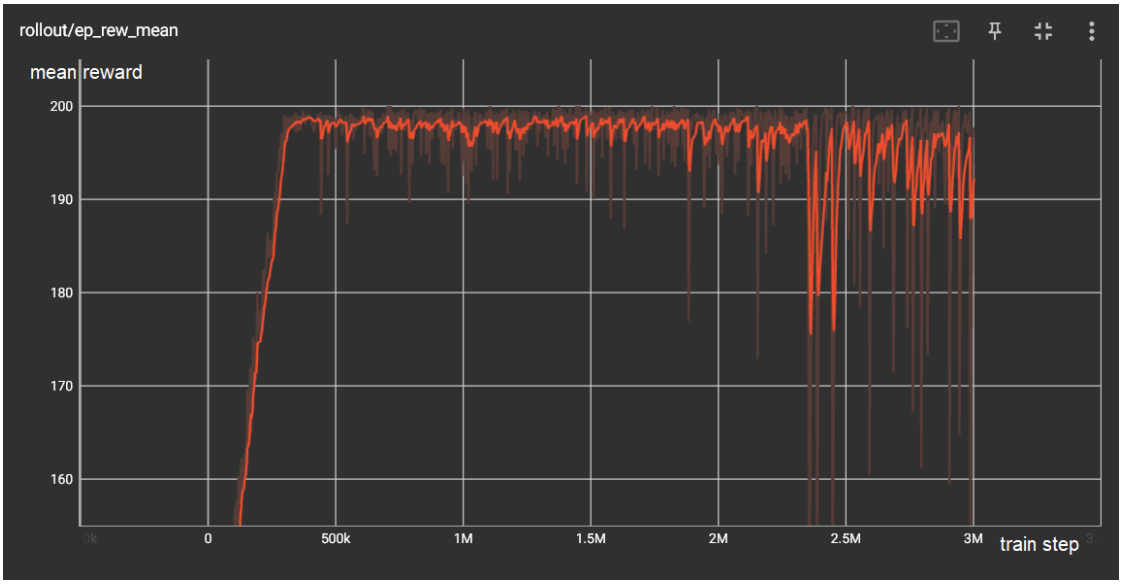
5.3 Gefinaliseerde PPO-agent

De gefinaliseerde PPO-agent werd getraind met de eerder ontdekte optimale hyperparameters. Figuur 10 weergeeft de gemiddelde episodelengte naarmate de agent traint. De gemiddelde episodelengte light rond de 8.



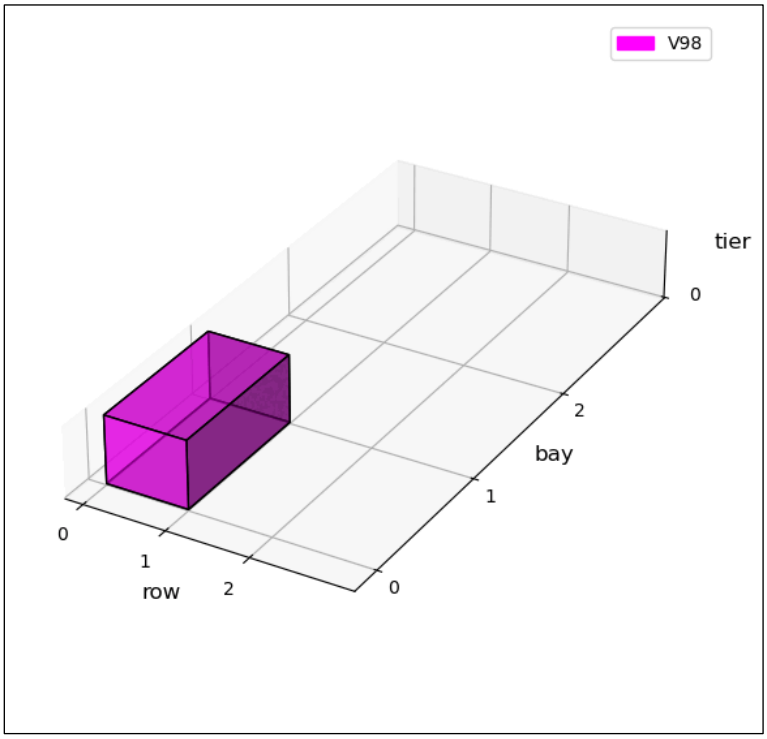
Figuur 10: de gemiddelde episodelengte per trainingsstap van het gefinaliseerd model

Figuur 11 weergeeft de gemiddelde episode reward naarmate de agent traint. Er is te zien dat de agent een plafond bereikt bij een gemiddelde episodereward van 200.



Figuur 11: de gemiddelde episodereward per trainingsstap van het gefinaliseerd model

Tot slot wordt de agent gebruikt om te voorspellen op welke locatie in het blok een container dient te worden geplaatst. Figuur 12 weergeeft de agent die containers plaatst op locaties in het blok voor 1 episode. Er is te zien dat containers met een overeenkomende bestemming naast elkaar worden geplaatst. Figuur 13 weergeeft de informatie zoals de uiteindelijke score, etc. nadat de episode afgerond is.



Figuur 12: voorspelde plaatsingen voor containers

```
step: 8
reward: 20
score: 200
observation: [[[1], [1], [1]], [[1], [0], [1]], [[1], [1], [1]]]
illegal moves: 0
done: True
```

Figuur 13: eindstand van voorspelde plaatsingen

6 Discussie

Na beide agents te hebben getraind, blijkt dat de gemiddelde episodelengte van PPO korter is dan A2C en PPO levert ook betere resultaten op dan A2C. Met A2C kunnen er problemen optreden waarbij een bepaalde trainingsroute een significante invloed heeft op de gewenste actie van de 'actor', waardoor deze minder goed is.

Daarnaast is er verschil te zien in hoe bepaalde hyperparameterwaarden de performance van de PPO-agent beïnvloede. Zo heeft learning_rate een grote spreiding voor verschillende waarden. Maar bij gae_lambda hebben verschillende waarden ongeveer dezelfde performance. Als er voor gae_lambda een non-optimale waarde gebruikt zou worden, zou dit minder invloed hebben vergeleken met een non-optimale waarde voor learning_rate. Niet elke hyperparameter heeft dezelfde hoeveelheid invloed.

Verder is de performance van de gefinaliseerd PPO-agent erg goed met een gemiddelde episodereward van 200. De environment biedt 9 locaties in totaal en 8 containers om te plaatsen. Elke goede container plaatsing ontvangt +20 als reward en +10 als volgend reward voor het naast elkaar plaatsen van containers met dezelfde bestemming. De theoretische beste score is namelijk 8 x 20 + 4 x 10 = 200. De agent behaald deze score ook. Dit komt ook overeen met de theoretische beste episodelengte van 8, omdat er 8 containers zijn om te plaatsen.

Kortom presteert de gefinaliseerde agent optimaal voor de environment en rewardfunctie waarmee het getraind is.

7 Conclusie

De hoofdvraag dat in dit project centraal stond, luidt als volgt “Hoe kan ervoor gezorgd worden dat containers op de kade op een efficiënte manier opgestapeld kunnen worden, zodat de afnemer van de containers hier makkelijk bij kan”. Voor het beantwoorden van deze hoofdvraag, is het probleemdomein van tevoren in kaart gebracht. Hieruit is gebleken dat de containerplaatsing op een kade momenteel handmatig verloopt, wat tijd en geld kost.

Om dit optimaliseringsprobleem op te lossen, is er gebruik gemaakt van RL. De toepassing van RL ging gepaard met twee verschillende agents, PPO en A2C, die interacties voerden met een environment. Naar aanleiding van evaluaties op de performance van de RL-agents, kwam PPO het best naar voren op basis van de episodelengte en de resultaten.

De optimale PPO-agent heeft er uiteindelijk voor gezorgd dat de containers op een efficiënte wijze worden geplaatst. Dit is bewerkstelligd door een container nauwkeurig te plaatsen aan de hand van de bestemming van een naastgelegen container. Het plaatsen van een container naast een ander container met een identieke bestemming, is afgeleid uit de ingerichte rewards en penalties in het environment.

Door gebruik te maken van de kracht en leervermogen van de PPO-agent, zorgt het voor besparen van tijd en geld. Het efficiënt plaatsen van containers in dit onderzoek, is in ieder geval een basis voor vervolgonderzoek voor in de scheepvaartindustrie.

Voor vervolgonderzoek zal de focus moeten liggen in de uitbreiding van de environment en de rewardfunctie. Een manier voor de uitbreiding van de environment is om een afmeting van 3x3x3 aan te houden, in plaats van 3x3x1. Voor de rewardfunctie is het van belang dat deze ook uitgebreid wordt naar het identificeren van gelijksoortige containerbestemmingen die op of onder elkaar zitten. Hierbij zal dan ook rekening gehouden moeten worden met de locaties die toegankelijk zijn voor een reachstacker. Bepaalde geplaatste containers kunnen er namelijk voor zorgen dat beschikbare locaties worden afgesloten. De uitbreiding van de environment naar de hoogte, het uitbreiden van de rewardfuncties en het bevorderen van de toegankelijkheid zal voor een meer praktische simulatie en weergave van de werkelijkheid zorgen.

8 Bibliografie

(sd).

Bick, D. (2021, augustus 15). *Towards Delivering a Coherent Self-Contained Explanation of Proximal Policy Optimization.* Opgehaald van https://fse.studenttheses.ub.rug.nl/25709/1/mAI_2021_BickD.pdf

Cofano. (sd). *Cofano | Software Solutions*. Opgehaald van Cofano: https://www.cofano.nl/nl/

Desaraju, N. (2022, 8 8). *Hands-On Introduction to Reinforcement Learning in Python*. Opgehaald van Medium: https://towardsdatascience.com/hands-on-introduction-to-reinforcement-learning-in-python-da07f7aaca88

Gupta, A. (2019, 6 7). *ML | Reinforcement Learning Algorithm : Python Implementation using Q-learning*. Opgehaald van GeeksforGeeks: https://www.geeksforgeeks.org/ml-reinforcement-learning-algorithm-python-implementation-using-q-learning/?ref=rp

indiamart.com. (sd). *Sany Srsc4535h 45 Ton Reach Stacker*. Opgehaald van India Mart: https://www.indiamart.com/proddetail/sany-srsc4535h-45-ton-reach-stacker-21624142730.html

Kansal, S., & Martin, B. (sd). *Reinforcement Q-Learning from Scratch in Python with OpenAI Gym*. Opgehaald van Learn Data SCI: https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/

Lewandowski, K. (2014, 12 31). CZECHOSLOVAK ACTIVITY TO PREPARE EUROPEAN NORMS FOR CONTAINERS BEFORE THE SECOND WORLD WAR. *Acta logistica*, pp. 1-7. doi:10.22306/al.v1i4.25

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., . . . Kavukcuoglu, K. (2016, June 16). *Asynchronous Methods for Deep Reinforcement Learning*. Opgehaald van Cornell University: https://arxiv.org/abs/1602.01783v2

Ntriankos, V. (2022). *Optimising the yard layout of Container Terminals.* Delft: TU Delft.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017, August 18). *Proximal Policy Optimization Algorithms*. Opgehaald van Cornell University: https://arxiv.org/abs/1707.06347

Shih-Chan, T., Jaw-Shen, W., Shen-Long, K., & Flor Melina, P. (sd). Categorized stacking models for import containers in port container terminals. *Maritime Economics & Logistics*.

Singh, A. (2018, 11). *Introduction to Reinforcement Learning*. Opgehaald van Datacamp: https://www.datacamp.com/tutorial/introduction-reinforcement-learning

Tewari, U. (2020, 04 15). Opgehaald van Medium.com: https://medium.datadriveninvestor.com/which-reinforcement-learning-rl-algorithm-to-use-where-when-and-in-what-scenario-e3e7617fb0b1

The Charthouse Group. (2022, 5 31). *Chapter 3.4 – Container Terminal Design and Equipment | Port Economics, Management and Policy*. Opgehaald van Port Economics, Management and Policy | A comprehensive analysis of the port industry: https://porteconomicsmanagement.org/pemp/contents/part3/container-terminal-design-equipment/

Verma, R., Saikia, S., Khadilkar, H., Agarwal, P., Shroff, G., & Srinivasan, A. (2019, 5 8). A Reinforcement Learning Framework for Container Selection and Ship Load Sequencing in Ports. *AAMAS '19: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2250-2252.