KTO KARATAY ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
ELEKTRİK VE BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
TEZLİ YÜKSEK LİSANS PROGRAMI

TAKTİK ÇEVRE SİMÜLASYON PROGRAMLARINDA SANAL VARLIKLARIN PEKİŞTİRMELİ ÖĞRENME İLE EĞİTİLMESİ

Fatiha Nur BÜYÜKOFLAZ

Yüksek Lisans Tezi

KONYA

Mart 2021

TAKTİK ÇEVRE SİMÜLASYON PROGRAMLARINDA SANAL VARLIKLARIN PEKİŞTİRMELİ ÖĞRENME İLE EĞİTİLMESİ

Fatiha Nur BÜYÜKOFLAZ

KTO Karatay Üniversitesi
Lisansüstü Eğitim Enstitüsü
Elektrik ve Bilgisayar Mühendisliği Anabilim Dalı
Yüksek Lisans Programı

Yüksek Lisans Tezi

Tez Danışmanı: Dr. Öğr. Üyesi H. Oktay ALTUN

Konya
Mart 2021

BİLDİRİM

Enstitü tarafından onaylanan Yüksek Lisans tezimin tamamını veya herhangi bir kısmını basılı veya dijital biçimde arşivleme ve aşağıda belirtilen koşullar dahilinde erişime açma iznini KTO Karatay Üniversitesine verdiğimi bildiririm. Bu izinle, Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak ve gelecekteki çalışmalar (makale, kitap, lisans, patent vb.) için tezimin tamamının veya bir bölümünün kullanım hakları yalnızca bana ait olacaktır.

Tezimin bütünüyle kendi çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Telif hakkı bulunan ve sahiplerinden yazılı izinle kullanılması zorunlu olan kaynakları, yazılı izin alarak kullandığımı ve istenildiğinde izinlerin suretlerini Üniversiteye teslim etmeyi taahhüt ederim.

Yükseköğretim Kurulu tarafından yayımlanan "Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge" kapsamında, tezim, aşağıda belirtilen koşullar haricince, YÖK Ulusal Tez Merkezi ve KTO Karatay Üniversitesi Açık Erişim Sisteminde erişime açılır.

☐ Enstitü / Fakülte Yönetim Kurulu kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren 2 yıl ertelenmiştir.[1]

☐ Enstitü / Fakülte Yönetim Kurulunun gerekçeli kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren ... ay ertelenmiştir.[2]

☐ Tezimle ilgili gizlilik kararı verilmiştir.[3][4]

19 Mart 2021

Fatiha Nur BÜYÜKOFLAZ

[1] MADDE 6(1) Lisansüstü tezle ilgili patent başvurusu yapılması veya patent alma sürecinin devam etmesi durumunda, tez danışmanının önerisi ve enstitü anabilim dalının uygun görüşü üzerine enstitü veya fakülte yönetim kurulu iki yıl süre ile tezin erişime açılmasının ertelenmesine karar verebilir.

[2] MADDE 6(2) Yeni teknik, materyal ve metotların kullanıldığı, henüz makaleye dönüşmemiş veya patent gibi yöntemlerle korunmamış ve internetten paylaşılması durumunda 3. şahıslara veya kurumlara haksız kazanç imkanı oluşturabilecek bilgi ve bulguları içeren tezler hakkında tez danışmanının önerisi ve enstitü anabilim dalının uygun görüşü üzerine enstitü veya fakülte yönetim kurulunun gerekçeli kararı ile altı ayı aşmamak üzere tezin erişime açılması engellenebilir.

[3] MADDE 7(1) Ulusal çıkarları veya güvenliği ilgilendiren, emniyet, istihbarat, savunma ve güvenlik, sağlık vb. konulara ilişkin lisansüstü tezlerle ilgili gizlilik kararı, tezin yapıldığı kurum tarafından verilir. Kurum ve kuruluşlarla yapılan işbirliği protokolü çerçevesinde hazırlanan lisansüstü tezlere ilişkin gizlilik kararı ise, ilgili kurum ve kuruluşun önerisi ile enstitü veya fakültenin uygun görüşü üzerine üniversite yönetim kurulu tarafından verilir. Gizlilik kararı verilen tezler Yükseköğretim Kuruluna bildirilir.

[4] MADDE 7(2) Gizlilik kararı verilen tezler gizlilik süresince enstitü veya fakülte tarafından gizlilik kuralları çerçevesinde muhafaza edilir, gizlilik kararının kaldırılması halinde Tez Otomasyon Sistemine yüklenir.

ETİK BEYAN

KTO Karatay Üniversitesi Lisansüstü Eğitim Enstitüsü Tez Hazırlama ve Yazım Kurallarına uygun olarak Dr. Öğr. Üyesi H. Oktay ALTUN danışmanlığında tarafımdan üretilen bu tez çalışmasında; sunduğum tüm veri, enformasyon, bilgi ve belgeleri bilimsel etik kuralları çerçevesinde elde ettiğimi, tüm değerlendirme, analiz, bulgu ve sonuçları bilimsel usullere uygun olarak sunduğumu, tez çalışmasında yararlandığım kaynakların tümüne bilimsel normlara uygun biçimde atıfta bulunarak kaynak gösterdiğimi, tezimin kaynak gösterilen durumlar dışında özgün olduğunu bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

19 Mart 2021

Fatiha Nur BÜYÜKOFLAZ

TEŞEKKÜR

Tez çalışmam sırasında kıymetli bilgi, birikim ve tecrübeleri ile bana yol gösteren ve destek olan değerli danışman hocam sayın Dr. Öğr. Üyesi Hüseyin Oktay ALTUN'a, ilgisini ve önerilerini esirgemeyen değerli hocam sayın Dr. Öğr. Üyesi Ali Öztürk'e teşekkür ve saygılarımı sunarım.

Bu tez çalışması HAVELSAN ve AutoDidactic Technologies şirketleri arasındaki işbirliği ile gerçekleştirilmekte olan FIVE-ML projesinde yapılan çalışmalar kapsamında yürütülmüştür. Projede görev almak suretiyle tezime doğrudan veya dolaylı olarak katkıları bulunan Esat Kalfaoğlu'na, Emre Fişne'ye, Cihan Çalışır'a, Furkan Ceran'a ve Ahmet Mutlu'ya teşekkürü borç bilirim.

Çalışmalarım boyunca destekleriyle her zaman hayatımı kolaylaştıran ve beni hiçbir zaman yalnız bırakmayan sevgili anneme ve babama sonsuz teşekkür ederim.

Fatiha Nur BÜYÜKOFLAZ
Mart-2021

ÖZET

Fatiha Nur BÜYÜKOFLAZ

Taktik Çevre Simülasyon Programlarında
Sanal Varlıkların Pekiştirmeli Öğrenme ile Eğitilmesi

Yüksek Lisans Tezi

Konya, 2021

Günümüzde uçuş eğitimlerinin büyük bir kısmı güvenlik gerekçeleri, gerçek araçların bakım, onarım ve kullanım maliyetleri, eğitim için gerekli olan savaş koşullarının gerçek dünyada zor oluşturulması gibi sebeplerden ötürü uçuş simülatörleri üzerinden gerçekleştirilmektedir. Bu uçuş simülatörleri genellikle taktik çevre yazılımları ile entegre çalışırlar. Pilot tarafından kontrol edilen yüksek gerçeklik sadakat derecesine sahip varlıklarla birlikte senaryoyu oluşturan diğer düşük gerçeklik sadakat derecesine sahip varlıklar bu taktik çevre simülasyon programları ile yönetilirler. Mevcut taktik çevre simülasyon programlarında ortamdaki varlıkların kontrolleri geleneksel kontrol dallanmaları ile uzman ekipler tarafından programlanmak suretiyle gerçekleştirilmektedir. Bu şekilde tasarlanan simülasyonlarda karşılaşabilecek bütün durum olasılıklarını düşünmek zaman almaktadır ve yazılımsal anlamda oldukça güçtür. Ayrıca simülasyonda kullanılan sanal varlıkların statik olarak tasarlanması oldukça sınırlı ve kendini tekrarlayan senaryolara olduğu gibi katı davranış modellerine de sebep olmaktadır. Bu ve benzeri problemler için taktik çevre sümülasyonlarındaki sanal varlıkların eğitilmesi ihtiyacı ortaya çıkmıştır. Bu varlıkların eğitilmesi için pekiştirmeli öğrenme algoritmalarından olan proksimal politika optimizasyonu tercih edilmiştir. Çalışmalar ticari bir taktik çevre simülasyon programı üzerinde gerçekleştirilmiştir. Sonuç olarak proje kapsamında kullanılan algoritmaların sanal varlıkları eğitmek için uygunluğu gösterilmiştir. Buna ek olarak birden çok sanal varlığın bulunduğu senaryolarda tüm varlıkların akıllandırılması suretiyle bir ekip olarak çalışabileceği ispatlanmıştır.

Anahtar kelimeler: Pekiştirmeli öğrenme, uçuş simülatörü, taktik çevre simülasyon programları, proksimal politika optimizasyonu, sanal varlıkların eğitilmesi

ABSTRACT

Fatiha Nur BÜYÜKOFLAZ

Training Virtual Entities with Reinforcement Learning
in Tactical Environment Simulation Programs

Master's Thesis

Konya 2021

Today, most of the flight trainings are carried out on flight simulations due to maintenance, repair and usage costs of real vehicles, security reasons and the fact that the war conditions required for training can be difficult in the real world. Flight simulators are usually integrated with tactical environment simulators. Along with the high fidelity model of an airplane controlled via a pilot, the rest of the low fidelity entities are controlled by this tactical environment simulators. In the existing simulation softwares, controlling agents in the tactical environment where the pilots are trained are performed by traditional computer coding techniques with many branches in order to account for many possibilities. It takes substantial time and effort to consider all the possibilities that may be encountered in simulation environments and it is a quite challenging task to program these entities. In addition, the static design of virtual entities used in simulation causes limited and repetitive scenarios and rigid behavioral models. For this problem and related problems, the need for intelligent virtual entities has emerged. Proximal policy optimization, which is a branch of reinforcement learning algorithm was preferred to train this agent. The studies were carried out on a highly sophisticated commercial simulation software. As a result, the algorithms used within the scope of the project have been proven to be suitable for smarting static virtual entities. In addition, it has been shown that virtual entities can work as a team with the proposed approach.

Keywords: Reinforcement learning, flight simulator, tactical environment simulation programs, proximal policy optimization, training virtual entities

İÇİNDEKİLER

KABUL VE ONAY i

BİLDİRİM ii

ETİK BEYAN iii

TEŞEKKÜR iv

ÖZET v

ABSTRACT vi

TABLOLAR DİZİNİ ix

ŞEKİLLER DİZİNİ x

SİMGELER DİZİNİ xiii

KISALTMALAR DİZİNİ xiv

1 GİRİŞ 1

2 İLGİLİ ÇALIŞMALAR 4

3 TAKTİK ÇEVRE SİMÜLASYONU 11

4 PEKİŞTİRMELİ ÖĞRENME 13

4.1 Pekiştirmeli Öğrenmeye Genel Bakış 13

4.2 Çalışma Prensibi 15

4.3 Bellman Denklemi 17

4.4 Pekiştirmeli Öğrenme Algoritmaları 18
4.4.1 Politika Tabanlı Öğrenme (Sadece Aktör) 18
4.4.2 Değer Tabanlı Öğrenme (Sadece Kritik) 19
4.4.3 Modele Göre Öğrenme 19
4.5 Q-Öğrenme 20
4.6 Soft Aktör Kritik 21
4.7 Proksimal Politika Optimizasyonu 23
4.7.1 PPO-Ceza 25
4.7.2 PPO-Kırpma 26

5 DENEY 28
5.1 Deney Kurgusu 28
5.1.1 Tasarım Unsurları 28
5.1.2 Donanım ve Ölçeklendirilebilirlik 29
5.1.3 Yazılım 30
5.1.3.1 Performans Metrikleri ve Wandb Kayıt Arayüzü 30
5.1.3.2 Algoritma 35
5.1.3.3 Temel Eylemler 40
5.2 Deney Sonuçları 43
5.2.1 3e3 Hava-Hava Senaryosu RL Eğitimi 43
5.2.2 Hava Yer Tahrip Sabit Kanat RL Eğitimi 46
5.2.3 Hava Yer Tahrip Döner Kanat RL Eğitimi 51

6 SONUÇ VE DEĞERLENDİRME 55

KAYNAKLAR 57

ÖZGEÇMİŞ 62

TABLOLAR DİZİNİ

5.1 Temel eylem kategorileri 41
5.2 Temel eylemler ve parametreleri 42

ŞEKİLLER DİZİNİ

3.1 Taktik çevre simülatörünün genel mimari yapısı 12

4.1 Varlık-çevre etkileşimi 16

4.2 Pekiştirmeli öğrenme yöntemleri 18

4.3 Modele göre pekiştirmeli öğrenme taksonomisi 20

5.1 Tasarımda yer alan unsurlar bütünü 29

5.2 Projede kullanılan aygıtlar arasındaki iletişim 29

5.3 Örnek Wandb kayıt arayüzü 33

5.4 Diğer ödül tasarımı örnekleri 35

5.5 Simülasyon ve sanal varlıklar arasındaki etkileşim 37

5.6 Sanal varlıkların eğitilmesinin ayrıntılı gösterimi 39

5.7 Python dilinde proksimal politika optimizasyonunun kullanımı 40

5.8 3e3 Hava-Hava senaryosunda eğitilen sanal varlıkların muhasım uçakları vurmaları sonucunda aldıkları ödül miktarının zamana göre değişimi. 44

5.9 3e3 Hava-hava senaryosunda eğitilen sanal varlıkların son 100 bölümden aldığı ortalama ödül değerinin zamana göre değişimi. 44

5.10 3e3 Hava-Hava senaryosunda eğitilen varlıkların kendilerini öldürme durumuna göre aldıkları ödül miktarı. Zaman içerisinde ölümler azalmakta, sanal varlıklar güzel bir öğrenme trendi yakalamaktadır. 45

5.11 Muhasım uçakları vurma sonucunda elde edilen ödül değerinin K-epoch=10 ve 50 için kıyaslanması. K-epoch artırıldığında, sanal varlıklar muhasım uçakları vurmayı daha hızlı öğrenmiştir. 46

5.12 Hava yer senaryosunda kurulan 4 aksiyon kümesi için 4 model yapısının ortak-ödül ve farklı-ödül durumunda karşı taraftaki hedeflerini vurabilme grafikleri. Y ekseni düşmanı vurma ödülüne karşılık gelmektedir. 48

5.13 Hava yer senaryosunda kurulan 4 aksiyon kümesi için 4 model yapısının ortak-ödül ve farklı-ödül durumunda tabi tutuldukları hava yer ödül ceza sistemine göre son 100 bölümden aldıkları ortalama skor grafiği. 48

5.14 Manevra modelinin ortalama skor grafiği. Biraz geç başlasa da öğrenme eğilimine girdiği gözlemlenmektedir. 49

5.15 Sensör aksiyonlarına karar veren modelin ortalama skor grafiği. Sert iniş çıkışlar eğitimi durdurup farklı değerlerle başlatma denemelerinden kaynaklıdır. Model ana öğrenme eğilimini büyük oranda korumaktadır. 49

5.16 Silah aksiyonlarına karar veren modelin ortalama skor grafiği. Burada silah aksiyonlarında cezada sert bir düşüşün yaşanmasının sebebi eğitimin silah ceza katsayısının artılırılıp devam ettirilmesinden kaynaklanan bir durumdur. Ceza artmış gibi gözükse de aslında toplamda cezanın azalmaya başlandığı gözlemlenmektedir. Dolayısıyla bu bir öğrenme kaybı değildir. Öğrenme sürecindeki silah eylemleri için oluşturulan ödül tasarımının değiştirilmesinden kaynaklanmaktadır. 50

5.17 Mesajlaşma aksiyonlarına karar veren modelin ortalama grafiği. Sert iniş çıkışlar eğitimi durdurup farklı değerlerle başlatma denemelerinden kaynaklıdır. Model ana öğrenme eğilimini büyük oranda korumaktadır. 50

5.18 Hedefi vurma ödülü, arkadaşını ya da kendini vurma cezası dışında ödül ceza almadan kaldığı yerden devam ettirilen modelin ortalama skor grafiği. Modelin iniş çıkışlara rağmen öğrenme trendine devam ettiği gözlemlenmiştir. 51

5.19 Eğitilen sanal varlığın zamana göre ortalama skor grafiği. 52

5.20 Eğitilen sanal varlığın zamana göre karşı hedefleri vurmadan aldığı ödül grafiği. Sanal varlık, ortalama skor grafiğindeki gibi zamanla daha iyi bir performans göstermeye başlamıştır. 53

5.21 Eğitilen sanal varlığın zamana göre dostunun ölümünün ceza grafiği. Sanal varlık başlangıçta dostuna karşı mühimmat atarken zamanla bu davranışı azaltmayı öğrendiği görülmektedir. 53

5.22 Eğitilen sanal varlıkların son 100 bölümden aldıkları ortalama ödül değerlerinin zaman adımı güncelleme parametrelerinin 1000 ve 4000 için kıyaslanması. Zaman adımı güncellemesi parametesi 1000 olduğunda eğitimin daha hızlı ilerlediği ama ilerleyen aşamalarında stabilizasyon problemleri ile karşılaştığı gözlemlenmektedir. 54

SİMGELER DİZİNİ

Simgeler	Açıklama
a	Aksiyon
$\hat{A}$	Avantaj fonksiyonu
d	Ortalama kestirim değeri
$\hat{g}$	Gradyan
r	Ödül
s	Durum
s'	Sonraki durum
V	Değer fonksiyonu
ε	Hiperparametre
γ	İndirim oranı
π	Politika

KISALTMALAR DİZİNİ

Kısaltmalar	Açıklama
ANN	Yapay sinir ağları (Artificial neural networks)
BT	Davranış ağacı (Behaviour tree)
CGF	Bilgisayar tarafından oluşturulan kuvvet (Computer generated force)
DQL	Derin Q öğrenme (Deep Q learning)
DQN	Derin Q ağı (Deep Q network)
DDPG	Derin deterministik politika değişimi (Deep deterministic policy gradient)
GA	Genetik algoritma (Genetic algorithm)
GP	Genetik programlama (Genetic programming)
LWACS	Hafif hava muharebe simülatörü (Light weight air combat simulator)
MADDPG	Çoklu aracılı derin belirleyici politika gradyanı (Multi-Agent Deep Deterministic policy gradient)
N-DOF	Serbestlik derecesi (Degree of freedom)
NLR	Hollanda Kraliyet Uzay Merkezi (Royal Netherlands Aerospace Centre)
PG	Politika değişimi (Policy-gradient)
PPO	Proksimal politika optimizasyonu (Proximal policy optimization)
RNN	Yinelenen sinir ağı (Recurrent neural network)
RL	Pekiştirmeli öğrenme (Reinforcement learning)
SAC	Soft aktör kritik (Soft actor critic)
TACSI	Taktik Simülasyon (Tactical simulation)
TRPO	Güven bölgesi politika optimizasyonu (Trust region policy optimization)

1 GİRİŞ

Uçak teknolojilerinin giderek daha çok önem kazandığı günümüzde, uçakların tasarımı kadar uçakları kullanacak olan pilotların eğitimi de büyük önem arz etmektedir. Pilotların eğitimini gerçek uçaklarla gerçekleştirmek hava sahası düzenlenmesi, yüksek maliyet, eğitim sırasında oluşabilecek riskler ve muhasımlar tarafından kullanılan savunma ve harp platformlarının gerçek dünyada tam olarak oluşturulmasının güçlüğü gibi nedenlerden dolayı oldukça zordur. Bu yüzden personel eğitimleri için uçuş simülasyonları sıkça tercih edilmektedir.

Pilot eğitimlerinde kullanılan uçuş simülasyonları genellikle taktik çevre simülasyonlarına entegre olarak çalışır. Taktik çevre simülasyonları vasıtasıyla, pilotun kontrol ettiği yüksek gerçeklik sadakat derecesine sahip (İng., high fidelity) varlık haricinde kalan ve senaryoyu oluşturan diğer düşük gerçeklik sadakat derecesine sahip varlıklar (İng., low fidelity) yönetilirler. Bu düşük sadekat derecesine sahip varlıklar yani hava, kara veya deniz yüzeylerindeki savunma ya da saldırı sistemlerini temsil eden özerk birimler bilgisayar tarafından kontrol edilen kuvvetler vasıtasıyla (İng. computer generated forces, CGF) oluşturulurlar.

CGF'ler, personelin görevlendirilmesi için hazırlık sürecinde, taktik eğitimlerde veya yeni strateji geliştirme aşamalarında kullanılır. CGF'ler her uygulama (veya her senaryo) için farklı programlanmaya ihtiyaç duyarlar. Geleneksel yöntemlerle oluşturulan bu kuvvetler, uyarlanabilir olmayan katı davranış modellerine sebep olur. Bu durum simülasyon eğitimi alan öğrencilerin, statik olarak programlanmış varlıklarla eğitim almasına ve eğitimlerinin zayıflamasına yol açar. Yeni senoryalara ihtiyaç duyulduğunda ise bunu gerçekleştirecek uzman personellere bağımlılık ortaya çıkar. Ayrıca senorya

oluşturma işlemi klasik kontrol dallanmaları ile gerçekleştirileceği için yeni senaryo oluşturma sürecinde bütün olasılıkları düşünmek ve kodlamak çoğu zaman mümkün değildir, veya mümkün olsa bile çok zahmetlidir. Bu nedenlerden, pilot adaylarının hem kendi görevlerini, hem de karşıt kuvvetlerin mevcut yeteneklerini ve taktiklerini simüle etmek için daha gerçekçi sanal ortamlara ve sürekli değişen dünyaya uyum sağlamak için yeni senaryolara artan bir ihtiyaç oluşmuştur.

Problem tanımında anlatılan sorunlara çözüm olarak bu çalışmada geleneksel senaryo odaklı modelleme yerine yapay zeka odaklı davanış modellemesine doğru bir geçiş önerilmiştir. Daha açık ifade edilmek istenirse simülasyonlardaki sanal varlıkları akıllandırma işlemi problem çözümü için kullanılacaktır. Yani sanal varlıklar, öğrenebilen dinamik sanal varlıklara dönüştürülecektir. Fakat sanal varlıkların eğitim süreçlerinde dikkate almaları gereken birçok durum ve etmen vardır. Öncelikle sensörleri ile algıladıkları çevrelerine uygun tepkiler vermeyi öğrenmeleri gerekmektedir. Daha sonra dostlarını ve muhasımlarını tanımalı ve bunların sınıf bilgilerine ve üzerlerinde taşıdıkları mühimmat tiplerine uygun şekilde hareket etmelidir. Dostlarıyla işbirliği içerisinde takım hareketleri sergileyebilmelidir.

Sanal varlıkların akıllandırılması için tercih edilen makine öğrenmesi yöntemi pekiştirmeli öğrenmedir (İng., Reinforcement learning, RL). Son yıllarda pekiştirmeli öğrenme geleneksel kontrol yöntemlerine kıyasla karmaşık ve öngörülemeyen kontrol problemlerini çözmenin yeni bir yolu olarak kabul edilmiştir. Nedeni ise RL algoritmasının sistem modelinden bağımsız olması ve kontrolör parametrelerinin algoritmanın kendisini yinelemesi ile optimize edilebilmesidir [6]. Bu sebeplerden ötürü pekiştirmeli öğrenme algoritması ilgili problem çözümünün kullanım için tercih edilmiştir.

Pilot adaylarının yaptığı seçimlerden öğrenerek bu seçimlere göre yeni stratejiler geliştiren bir simülasyonun, pilot eğitimlerini çok daha farklı bir noktaya taşıması öngörülmektedir. Proje tamamlandığında alanlarında daha donanımlı ve uzman savaş

pilotlarının yetiştirilmesine olanak sağlayacak yeni bir sistem tasarlanacaktır. Var olan kısır senaryo sistemi kendini yenileyebilen bir sisteme evrilecektir. Ayrıca emek yoğun bir senaryo mekanizmasından, hesaplamalı senaryo otomasyon mekanizmasına geçilecektir.

2 İLGİLİ ÇALIŞMALAR

Donanım teknolojilerindeki hızlı ilerlemeler ve pekiştirmeli öğrenme algoritmalarındaki gelişmeler, simülatörlerdeki sanal varlıkların ortamla etkileşimden elde edilecek tecrübelerle akıllandırılabileceği ihtimalini gündeme getirdi. Geleneksel simülasyon teknolojilerinde, sanal varlıkların davranışı, belirli bir senaryo üzerinde önceden belirlenmiş eylemleri öngören komut dosyaları tarafından yönetilmektedir. Komut dosyalarının hazırlanması ise yüksek karmaşıklık, katı ve gerçekçi olmayan davranışlar gibi problemleri beraberinde getirir. Bu problemlerin üstesinden gelmek için literatürde çeşitli çalışmalar yapılmıştır. Biz bu bölümde, bizden önce yapılan çalışmaları özetleyerek, kendi çalışmamızı literatürde konumlandırmaya çalışacağız.

Özellikle savaş uçağı eğitim simülatörlerinde pilotları, it dalaşı olarak da bilinen birebir mücadelede havada üstünlük kurma konusunda eğitmek önemli bir misyondur. Bu konuda insan kontrolündeki simülatör ile bilgisayar tarafından kontrol edilen sanal uçağın kapışması ne kadar gerçekçi olursa, eğitim misyonu o denli amacına ulaşmış olacaktır. Bu konuda yapılan çalışmaların ilki sayılabilecek çalışma, 2012 yılında gerçekleştirimiştir (Teng, Tan, Ong, & Lee, 2012). Bu çalışmada savaş pilotlarının simülatör temelli eğitimi için adapte olabilen karakteristiğe sahip bilgisayar kontrollü bir sanal uçağın, kural tabanlı bilgisayar kontrollü bir sanal uçaktan daha etkili olabileceğini göstermek amacı ile bir çalışma yayınlanmıştır. Öğrenen sanal varlık, FALCON olarak adlandırılan kendi kendini organize eden bir sinir ağı tarafından yönlendirilir. Kullanılan sinir ağı uyarlanabilir rezonans teorisine (İng., adaptive resonance theory, ART) dayanarak, durumları aşamalı olarak öğrenebilir ve genelleştirebilir. Ayrıca pekiştirmeli öğrenmeyi kullanarak çevre ile gerçek zamanlı etkileşimler sırasında bilgi

keşfedebilir.

Çalışmada akıllı sanal varlıkların performansını değerlendirmek için çok uluslu bir simülatör üreticisi ile işbirliği gerçekleştirilmiştir. Ticari sınıf simülasyon platformu kullanılarak iki ayrı denek grubundan oluşan (iki stajyer pilot ve üç deneyimli pilot) bir deney gerçekleştirilmiştir. Akıllandırılmış sanal varlığın stajyer pilotlara ve deneyimli pilotlara gösterdiği ilk zorluklara bakılarak, beklenmedik manevralarla rakibin modellenmesi için, kural tabanlı sanal varlığa göre daha uygun olduğu gösterilmiştir. Öte yandan, deneyimli pilotlar tarafından akıllı sanal varlıkların füze savaşı senaryolarına iyi uyum sağlamadığı fark edilmiştir.

J. Roessingh ve arkadaşları tarafından 2011 yılında hazırlanan raporda savaş pilotları için simülasyon eğitiminin geliştirilmesi amacı ile gerçekçi taktik davranış sergileyen sanal kuvvetlerin yapay zeka teknikleri ile nasıl sentezlenebileceği ele alınmıştır. Havadan havaya bir görevde yer alan akıllı CGF'lerin ortaya çıkan tüm taktik sorunlarını çözmek için tek bir tekniğin olmadığı sonucu çıkarılmış ve hibrit bir sistem önerilmiştir. Bu hibrit çözüm pekiştirmeli öğrenme ve yapay sinir ağından oluşmaktadır. Simülasyon gibi karmaşıklığı yüksek ortamlarda istenen yanıtlar genellikle bilinmediği için pekiştirmeli öğrenme, her denemede yanıtları iyileştirmek amacı ile tercih edilirken, sinir ağı ise pekiştirmeli öğrenme sırasında oluşan durum-eylem uzayına yaklaşmak için kullanılmıştır (Roessingh et al., 2011).

J. Roessingh, A. Toubman ve arkadaşlarının 2015 yılında yayınladıkları bir diğer makalede daha önceki çalışmalarında füze vuruşlarının şansa bağlı olması nedeniyle bazı durumlarda uygunsuz bir şekilde ödüllendirilmiş ve cezalandırılmış CGF'ler ele alınmıştır. Ödül işlevinde bu öldürme olasılığını hesaba katmanın performansı artıracağı düşünülmüş, sahte ödülleri ve cezaları gidermek için yeni bir ödül işlevi önerilmiştir. Testler, sanal varlıkların eylemlerinin beklenen sonucuna göre ödüllendiren bu işlevin kullanımının, çeşitli senaryolarda CGF'lerin performansını önceki ödül işlevine kıyasla

önemli ölçüde artırdığını göstermektedir (Toubman, Roessingh, Spronck, Plaat, & Van Den Herik, 2015).

Savaş uçakları için simülasyon eğitimini daha etkili hale getirmek amacı ile yapılan çalışmalardan bir diğeri 2019'da J. Källström ve arkadaşı tarafından yayınlanmıştır. Bu çalışmada yapıcı simülasyonlar için gelişmiş, uyarlanabilir davranış modelleri oluşturma sürecini otomatikleştirmek için makine öğrenimi tekniklerinin nasıl kullanılabileceği ele alınmıştır. Bir dizi ilk deney sonucunda pekiştirmeli öğrenmenin, özellikle birden fazla sanal varlıklı ve çok amaçlı derin pekiştirmeli öğrenmenin, sentetik pilotların hava muharebe senaryolarında çatışan hedefler arasında işbirliği yapmayı ve önceliklendirmeyi öğrenmesine olanak tanıdığı gösterilmiştir. Sonuçlar umut verici olsa da, hava muharebe simülasyonunun karmaşık alanına tamamen hakim olmak için daha fazla algoritma geliştirmenin gerekli olduğu sonucuna varılmıştır (Källström & Heintz, 2019).

J. Källström ve arkadaşları tarafından 2020'de yayınlanan devam çalışmasında ise hava muharebe simülasyonunda koordinasyonu öğrenmek için çok sanal varlıklı derin takviye öğreniminin uygulamaları incelenmiştir. Uygulama alanı kapsamında takviye öğrenimi için üç ana kullanım durumu belirlenmiştir. Bu durumlara bağlı olarak ise iki ayrı senaryo ve senaryolara özgü deney tasarımları gerçekleştirilmiştir. Birinci senaryoda sanal varlıkların seyrek ödüllere ve geniş arama alanına rağmen öğrenmelerine yardımcı olmak için müfredatla öğrenme kullanılmıştır. İkinci senaryoda ise çok varlıklı derin belirleyici politika gradyanı (İng., multi-agent deep deterministic policy gradient, MADDPG) ve derin deterministik politika değişimi (İng., deep deterministic policy gradient, DDPG) yöntemleri kullanılmıştır. Müfredat öğreniminin hava muharebe alanının karmaşıklığını ele almak için umut verici bir yaklaşım olduğu, davranışları öğrenme tamamlandıktan sonra da ayarlanabilen çeşitli özelliklere sahip sanal varlıklar oluşturmak için ise çok amaçlı öğrenmenin umut verici bir yaklaşım olduğu

gösterilmiştir (Källström & Heintz, 2020).

Uçuş simülatörlerinin kontrol sistemini pekiştirmeli öğrenme kullanarak geliştirme üzerine 2019 yılında bir çalışma gerçekleştirilmiştir. Bu çalışma kapsamında serbestlik derecesi(İng., degree of freedom, 2-DOF) uçuş simülatörünün denetleyicisini tasarlamak için derin Q ağı (İng., deep Q network, DQN), politika-değişimi (İng., policy-gradient, PG) ve DDPG olmak üzere üç modelsiz RL yöntemi kullanılmıştır. Kullanılan yöntemlerin analizi sonucunda ise DQN algoritmasının ayrık görevler için diğer iki yöntemden daha uygun olduğu kanıtlanmıştır. PG ve DQN'nin avantajlarını birleştiren DDPG algoritmasının ise karmaşık görevlerde performansının çok iyi olduğu gösterilmiştir. Ayrıca modelsiz pekiştirmeli öğrenme algoritmalarının yeterli veri elde etmesi için sanal varlığın, sistem bir yakınsama durumuna ulaşana kadar ortamı sürekli olarak araştırması gerektiği gözlenmiştir. Buna bağlı olarak veri verimsizliğinin kaçınılmaz bir son olduğu ve algoritmanın verimliliğinin nasıl artırılacağının çözülmesi gereken birincil problem olduğu vurgulanmıştır (Zuo, Deng, Yang, & Huang, 2019).

2018 yılında yayınlanan yüksek lisans tezinde havadan havaya simülasyonlarda it dalaşı senaryosunun pekiştirmeli öğrenme ile nasıl geliştirilebileceği ele alınmıştır. Çalışmada simülator olarak Saab AB tarafından üretilen taktik simülasyon (İng.,Tactical Simulation, TACSI) kullanılmıştır. İlgili senaryo için pekiştirmeli öğrenme algoritmalarından biri olan genetik algoritma (İng., genetic algorithm, GA) ile yapay zeka tekniklerinden olan genetik programlama (İng., genetic programming, GP), simülasyon ortamında çözüm alanını araştırmak ve yapay zeka davranışları geliştirmek için bir optimizasyon algoritması olarak kullanılmıştır. Genetik programlama tekniklerinin kombinasyonu ile Davranış ağacı (İng., behavior trees, BT) adı verilen bir çerçeve sunulmuştur. Eğitimler sonucunda BT çerçevesinin, koşulları sürekli olarak kontrol etmek ve her tik için ağacı geçmek için çok fazla hesaplama gücü kullanıldığı görülmüştür. Bunu değiştirmek için, BT yerine ağaç geçişlerinin tekrarlanmasını azaltan bellek düğümlerinin daha etkili ola-

bileceği öngörüsünde bulunulmuştur (Ted, 2018).

Hollanda Kraliyet Uzay Merkezi (NLR) çatısı altında başlayan akıllı haydutlar (İng., smart bandits), davranış modellemesi için iyileştirilmiş yöntemler arayan çok sayıda doktora öğrencisinin katkısıyla oluşturulmuş bir araştırma projesidir. Proje kapsamında 2018 yılında yayınlanan çalışmada geleneksel modelleme teknikleri ile oluşturulan CGF'lerin davranışsal yeteneklerinin oldukça temel olduğu belirtilmiştir. Bu sebeple de davranış modellemesinde, CGF'lerin geleneksel senaryo odaklı modellemeden daha temsilci odaklı modellemeye doğru giden bir paradigma kaymasının gerekliliği vurgulanmıştır. Bu bağlamda simüle edilmiş kuvvetlerin davranışını modellemeye yönelik bir yapay zeka çözümü sunulmuştur. (Centre, 2018).

Uçuş simülatöründe RL'nin sentetik düşmanları iyileştirmek amacı ile nasıl kullanılabileceğini keşfetmek için 2020 yılında bir tez çalışması gerçekleştirilmiştir. Bu çalışmada TACSI simülatörünün nasıl davrandığını anlamak ve sanal varlığın bilgisayar kontrollü sentetik bir düşmanı yenmesini sağlamak için RL algoritmalarından biri olan Q-öğrenme kullanılmıştır. Çalışma sonucunda RL alanındaki mevcut Q-öğrenme yaklaşımlarının ümit verici olduğu ve bazı uygulamalarda şaşırtıcı sonuçlar verdiği söylenmiştir. Fakat yapay zekanın davranışını eğitmek için gereken eğitim süresinin az olması ve bazı kısıtlı çalışma şartları sebebi ile eğitilen sanal varlığın, bir uzman tarafından yapılan davranışlar kadar iyi performans sergileyemediği sonucuna ulaşılmıştır (Lindberg, 2020).

A. Toubman tarafından 2020 yılında tamamlanan doktora tezinde, hava muharebe eğitim simülasyonlarında sanal rakipler için davranış modellerinin makine öğrenimi yardımı ile ne ölçüde otomatik olarak oluşturulabileceği araştırılmıştır. Araştırma kapsamında beş ayrı problem üzerine yoğunlaşılmış ve çözümleri için farklı metodolojiler ile birlikte farklı simülasyonlar kullanılmıştır. Birinci problem olarak ekip koordinasyonu sağlayan hava muharebe davranış modelleri hangi ölçüde oluşturulabilir sorusu ele alınmıştır.

Bu problem çözümü için ulaşılan sonuç ise dinamik komut dosyası aracılığıyla, birden fazla takım davranış biçimi oluşturulabileceği ve ekip üyelerinin üstlendiği rollerin bu şekilde kolay incelenebileceğidir. İkinci problem olarak hava muharebesinde CGF'ler için ödül işlevi ne ölçüde iyileştirebilir sorusu üzerine yoğunlaşılmıştır. Ödül işlevini iyileştirmek için domain-reward ve AA-reward adında iki yeni ödül işlevi geliştirilmiştir. Domain-reward CGF'lerin performansını iyileştirmede başarısız olurken, AA-reward kullanılması nihai performansta %12,6 artışa yol açmıştır.

Çalışmada üçüncü problem ise dinamik komut dosyaları ile oluşturulan bilginin, farklı senaryolarda CGF'ler arasında ne ölçüde başarıyla aktarılabileceğidir. Bu araştırma sorusuna cevap bulabilmek amacı ile hava muharebe simülasyonlarında transfer öğrenimi için bir kullanım senaryosu tasarlanmıştır. Sonuç olarak ise dinamik betikleme ile oluşturulan bilginin büyük ölçüde başarıyla aktarılabileceğine ulaşılmıştır. "Eğitim simülasyonlarında kullanılmak üzere makine tarafından üretilen hava muharebe davranış modellerini nasıl doğrulamalıyız?" sorusu dördüncü problem olarak ele alınmıştır. Makine tarafından üretilen davranış modellerinin doğrulanması için tek bir çözüm olmadığı için araştırma sorusuna cevap olarak beş adımdan oluşan yeni bir doğrulama prosedürü geliştirilmiştir. Beşinci ve en son problem olarak ele alınan sorun ise dinamik komut dosyası ile oluşturulan hava muharebe davranış modelleri eğitim simülasyonlarında kullanım için ne ölçüde geçerli olduğu sorusudur. Çözüm için bir önceki adımda oluşturulan doğrulama prosedürü savaş jeti simülasyonlarında, aktif görev yapan eğitmen pilotları tarafından gerçekleştirilmiştir.

Değerlendirme sonucunda ise dinamik komut dosyası aracılığıyla oluşturulan hava muharebe davranış modellerinin, eğitim simülasyonlarında kullanım için orta derecede geçerli olduğu çıkarımında bulunulmuştur. Çalışmada 1, 2 ve 3'üncü soruları yanıtlamak için hafif hava muharebe simülatörü (İng., light weight air combat simulator, LWACS) kullanılırken 5. soruya yanıt aramak için ise Hollanda Havacılık ve Uzay

Merkezi NLR'nin Fighter 4-Ship simülatörü kullanılmıştır. Çalışmada çıkarılan nihai sonuç ise dinamik komut dosyası oluşturmanın hava muharebe davranış modellerinin otomatik olarak üretilmesini büyük ölçüde kolaylaştırdığı ve zorluklara cevap verecek kadar esnek olduğudur (Armon, 2020).

Bu çalışmayı daha önce yapılan benzer çalışmalardan farklı kılan özellik kullanılan simülasyon ve tercih edilen pekiştirmeli öğrenme algoritmasıdır. Literatür incelendiğinde simülasyonlardaki sanal varlıkları akıllandırmak için farklı metodolojilerin ve algoritmaların kullanıldığı gözlemlenmiştir. Fakat çalışma kapsamında kullanılan poroksimal politika optimizasyonun taktik çevre simülasyonlarını akıllı hale getirmek için kullanıldığı başka bir çalışma ile karşılaşılmamıştır.

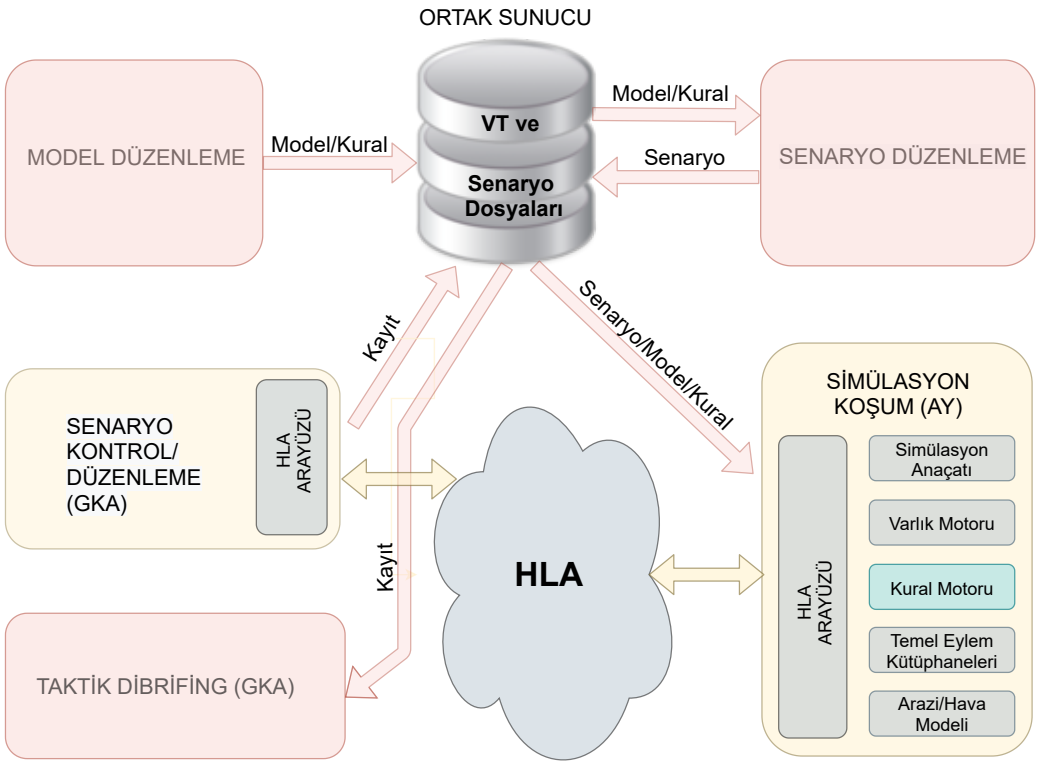
3 TAKTİK ÇEVRE SİMÜLASYONU

Tez kapsamında kullanılan ve pekiştirmeli öğrenme algoritmaları ile içindeki varlıkları akıllandırdığımız simülasyon ticari bir taktik çevre simülasyonu yazılımıdır. Bu yazılım, muharip platformlara ait simülatörler için çeşitli senaryolar geliştirilerek, taktik eğitimlerin etkin bir şekilde uygun maliyet ile verilebilmesini sağlamak amacı ile üretilmiş bir simülasyondur. Bu yazılım sayesinde hava, kara ve deniz platformlarının davranışları çeşitli harp ekipmanlarını içerecek şekilde sanal olarak modellenebilmekte ve böylelikle simülatörler için gerekli taktik ortamlar sağlanabilmektedir.

Simülasyondaki komut dosyalarını oluşturmak için tercih edilen dil Python'dır. Dağıtık mimari kullanılarak tasarlanan yazılımın genel yapısı Şekil 3.1'de verilmiştir. Şekilde çevrim dışı olarak çalışan model düzenleme, senaryo düzenleme ve taktik dibrifing sistemleri pembe olarak gösterilmiştir. Bu alt sistemlerden ilki olan model düzenleme modülünde taktik çevre yazılımı envanterinde bulunan farklı sistemler (örn. tanklar, avcı uçakları ve deniz altları vb.), bu sistemlerin kullandığı teçhizatlar (örn. silahlar, sensörler ve haberleşme cihazları vb.) ve sistemlerin farklı senaryolarda gerçekleştirecekleri davranış şekilleri modellenmektedir. İkinci modül olan senaryo düzenleme modülünde eğitim isterleri doğrultusunda gerekli olan sanal harekat ortamının oluşturulabilmesi için mevcut kurallar ve sistemlerin senaryo içindeki davranışları ilişkilendirilir.

Şekil 3.1'de sarı olarak gösterilen modüller ise senaryoların gerçek zamanlı olarak çalıştırıldığı ve izlendiği kısımlardır. Simülasyon koşum modülü, üzerinde hazırlanan senaryolarom çalıştırıldığı modüldür. Kendi içinde beş alt birimden oluşan modülde, anaçatı tüm alt birimlerin senkronize bir şekilde gerçek zamanlı olarak çalışmasını

sağlar. HLA arayüzü, modülün dış arayüz birimi olarak görev yapar. Mavi renkte gösterilen kural motoru Python dili ile tanımlanan kodları yorumlayarak, eylemi gerçekleştirir ve varlığın bir sonraki aşamada göstereceği davranışları (örn. ateş etme, kaçma, saldırı yapma vb.) belirler. Proje kapsamında değişiklik yapılacak birim burasıdır. Kural tabanlı olarak çalışan bu birim yapay zeka tabanlı çalışan bir sisteme dönüştürülecektir. Diğer üç alt birim ise senaryo içindeki varlıkların fiziksel simülasyonunu gerçekleştirir.



Şekil 3.1: Taktik çevre simülatörünün genel mimari yapısı

4 PEKİŞTİRMELİ ÖĞRENME

4.1 Pekiştirmeli Öğrenmeye Genel Bakış

Pekiştirmeli öğrenmenin tarihi, yıllar içinde iç içe geçmeden önce bağımsız olarak takip edilen uzun ve zengin iki ana konu üzerinde ilerlemiştir. İlk konu optimum kontrol problemi ve değer fonksiyonları ile bunların dinamik programlama kullanılarak çözümü ile ilgilidir. Optimum kontrol terimi, zaman içindeki dinamik sistem davranışının ölçüsünü en aza indirmek amacı ile denetleyici tasarlama sorununu tanımlamak için kullanılmaya başlanmıştır. Bu soruna yönelik yaklaşımlardan biri 1950'lerin ortalarında Richard Bellman ve arkadaşları tarafından on dokuzuncu yüzyılda geliştirilen Hamilton ve Jacobi teorisinin genişletilmesiyle ortaya çıkmıştır. Bu yaklaşım, günümüzde Bellman denklemi olarak adlandırılan, dinamik bir sistemin durumu ve bir değer fonksiyonu veya "optimal dönüş fonksiyonu" kavramlarının kullanıldığı işlevsel bir denklemi tanımlar. Bellman ayrıca 1957 yılında Markov karar süreçleri olarak bilinen optimal kontrol probleminin ayrık stokastik versiyonunu tanıtmışıtır. 1960'a gelindiğinde ise Ron Howard Markovian karar süreçleri için politika yineleme yöntemini tasarlamıştır. Bunların tümü, modern pekiştirmeli öğrenmenin teorisi ve algoritmalarının altında yatan temel unsurlardır (Sutton & Barto, 2018).

İkinci ana konuysa, deneme yanılma yoluyla öğrenmeyle ilgilidir ve "pekiştirme" teorilerinin yaygın olduğu psikoloji dalında başlamıştır. Deneme yanılma yoluyla öğrenmeyi kısa ve öz bir şekilde ilk ifade edenlerden biri Edward Thorndike'dir. Thorndike pekiştirici olayların eylem seçme eğilimi üzerindeki etkisini açıkladığı için bu durumu "etki yasası" olarak adlandırmıştır. Etki yasası arama ve hafızayı birleştirmenin temel

bir yolu olarak ifade edilebilir. Bu birleştirme ile her durumda birçok eylem arasından deneme ve seçme şeklinde arama yapılır ve hangi eylemlerin en iyi sonuç verdiği hatırlanır. Daha sonra bu eylemlerin en iyi oldukları durumlar hafızada ilişkilendirilir. Arama ve hafızayı bu şekilde birleştirmek, öğrenmeyi pekiştirmek için çok önemlidir (Sutton & Barto, 2018). Yukarıda belirtilen bu iki ana konunun, yani mühendislikteki optimal kontrol teorisi ve psikolojideki etki kanununun bir araya gelmesi ile pekiştirmeli öğrenme ortaya çıkar (Sutton & Barto, 1999).

Deneme yanılma ile öğrenmenin kullanıldığı en eski araştırmalardan biri 1954 yılında M. Minsky tarafından yazılan doktora tezidir. Nöral Analog Pekiştirme Sistemleri Teorisi ve Beyin Modeli Problemine Uygulanması adlı bu tezde Minsky pekiştirmeli öğrenmenin hesaplamalı modellerini tartışmıştır. Sinir ağlarında öğrenme, ikincil pekiştirme ve sinaptik modifikasyonlar hakkında birçok yeni teori öne sürmüştür. Ayrıca ilk rastgele kablolu sinir ağı öğrenme makinesi olan spatial-numerical association of response codes'un (SNARC) yapımını bu çalışmada anlatmıştır (Minsky, 1954).

Pekiştirmeli öğrenmenin kullanıldığı ilk çalışmalardan bir diğeri 1963 yılında Donald Michie tarafından yapılan oyun Öğrenmenin Mekanizasyonu Üzerine Deneyler adlı çalışmadır. Bu çalışmada Michie, Tic-Tac-Toe oyununu oynaması için deneme-yanılma yöntemi kullanarak matchbox educable naughts ve crosses engine (MENACE) adında bir sistem tasarlamıştır. Bu sistem bir bilgisayarın herhangi bir görevde iyi olmak için ödülden ve cezadan öğrenebileceğini göstermiştir (Michie, 1963).

1973 yılında Widrow ve arkadaşları başarı ve başarısızlık sinyallerinden öğrenebilen yeni bir takviye öğrenme biçimi tasarlamış ve oluşturdukları yönteme eleştirmenle öğrenme adını vermişlerdir. Ayrıca yaptıkları çalışmada eleştirmenle öğrenme yönteminin yirmi bir oyununu oynamayı nasıl öğrenebileceğini göstermişlerdir (Widrow, Gupta, & Maitra, 1973).

Sutton ve Barto ise 1980'li yıllarda sürdürdükleri farklı çalışmalar ile aktör-eleştirmen

mimarisinde zamansal fark öğrenimini kullanmak için yeni bir yöntem geliştirmişlerdir. Daha sonra bu yöntemi direk dengeleme problemine uygulayarak önemli başarılar elde etmişlerdir (Barto, Sutton, & Anderson, 1983).

Pekiştirmeli öğrenme ile ilgili dikkat çeken bir başka uygulama Gerry Tesauro'un 1992 yılında geliştirdiği TD-Gammon'dur. TD-Gammon yalnızca kendisine karşı oynayarak ve sonuçlardan öğrenerek tavla oynamayı kendi kendine öğrenebilen bir sinir ağıdır. Ortaya çıktığı dönemde, zamanın en iyi insan tavla oyuncularının biraz altında bir oyun seviyesine ulaşacak kadar büyük bir başarıya imza atmıştır (Tesauro, 1995).

Günümüzde ise pekiştirmeli öğrenme robotik (Kober, Bagnell, & Peters, 2013), otonom arabalar (Fayjie, Hossain, Oualid, & Lee, 2018), insansız hava araçları (Bagnell & Schneider, 2001), simülasyonlar (Mahadevan, Marchalleck, Das, & Gosavi, 1997; Wiering, Vreeken, van Veenen, & Koopman, 2004), strateji oyunları (Wender & Watson, 2012) gibi karmaşık ve öngörülemeyen kontrol problemleri için en çok tercih edilen makine öğrenmesi yöntemlerinden biri olmuştur. SpaceX tarafından geliştirilen Falcon 9 roketinin dünyaya geri inerek tekrar kullanılabilir olmasını sağlamak için pekiştirmeli öğrenme tercih edilmesi şu anki kullanımına güzel bir örnektir.

4.2 Çalışma Prensibi

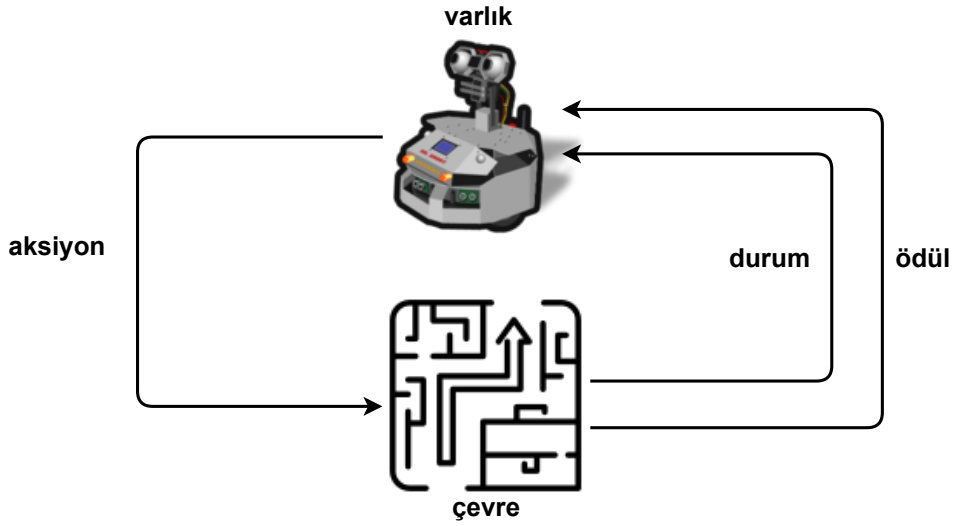
Makina öğrenmesinin, öğrenme yöntemine göre üç alt dalından biri olan RL'nin diğer yöntemlerden temel farklarından birisi öğrenme işlemini deneme yanılma yoluyla gerçekleştirmesidir. Bu denemeler sonucunda elde edilen ödül ve cezaların etkileri hemen ortaya çıkmaz gecikmeli bir öğrenme işlemi söz konusudur. RL bulunduğu ortam ile etkileşim kuran varlığın karşılaştığı durumlarda seçtiği aksiyonlara karşılık elde ettiği toplam ödülü en çoklama esasına dayanır. Pekiştirmeli öğrenmede kullanılan bazı temel terimlere aşağıda değinilmiştir.

- Varlık, bulunduğu ortamda başlangıçta rasgele seçtiği eylemleri zaman içinde de-

neyimleyerek öğrenen etmendir.

- Çevre, bir veya daha fazla varlığın ödül ve cezaya dayalı öğrenme işlemini gerçekleştirdiği dinamik de olabilen ve kendine özgü aksiyonlar içeren eylem alanıdır.
- Aksiyon, çevreye özgü olarak önceden tanımlanmış olan ve varlığın gerçekleştirebileceği tüm olası hareketlerdir.
- Durum, varlığın içinde bulunduğu çevrede karşılaşabileceği tüm olası halleri temsil eden gösterimdir.
- Ödül, varlığın çevrede o an bulunduğu durumda gerçekleştirdiği aksiyona bağlı olarak elde ettiği değerdir.

Pekiştirmeli öğrenme ile ilgili yukarıda verilen temel terimlerin birbirleri ile etkileşimi Şekil 4.1'de gösterilmiştir.



Şekil 4.1: Varlık-çevre etkileşimi

Şekil 4.1'e göre, varlık çevrede algıladığı duruma karşılık toplam ödülü en üst düzeye

çıkaracak aksiyonu seçer. Seçtiği aksiyona karşılık gelen anlık ödülü alarak öğrenme işlemine devam eder. Varlık çevre etkileşimi gerçek hayatla ilişkilendirerek açıklamak istenirse bir öğrencinin yıl boyu gösterdiği performans örnek olarak verilebilir. Öğrenci eğer verimli bir çalışma gerçekleştirirse yıl boyu kendisi için gerekli olan bilgileri öğrenerek yıl sonunda başarılı bir sonuç elde eder. Fakat düzenli ve verimli çalışmadan uzak bir yıl geçirirse, yıl sonunda başarılı bir performans elde edemeyeceğini deneyimleyerek öğrenmiş olur.

4.3 Bellman Denklemi

1957 yılında Richard E. Bellman tarafından yapılan çalışma ile yayınlanan (Bellman, 1957) Bellman denklemi karmaşık optimizasyon problemlerini çözmek için kullanılan bir yöntemdir. Bu yöntem optimizasyon problemini farklı zaman aralıklarında daha küçük problemlere bölerek her problemi kendi içinde çözme esasına dayanır. Bellman denklemi RL'de değer fonksiyonu olan $V(s)$'nin bir sonraki durum olan $s^{'}$ daki değeri $V(s^{'})$ ile ilişkisini ifade etmek için kullanılır.
RL'de deterministik (sanal varlığın bir eylemi gerçekleştirmesinin kesin olması) bir ortam için Bellman Denklemi, Denklem 4.1'de verilmiştir.

$$V(s) = \max_{a}(R(s,a) + \gamma V(s^{'})) \tag{4.1}$$

Denklem 4.2'de ise stokastik (sanal varlığın eylemi gerçekleştirmesinin olasılıklara bağlı olması) ortam için Bellman Denklemi verilmiştir.

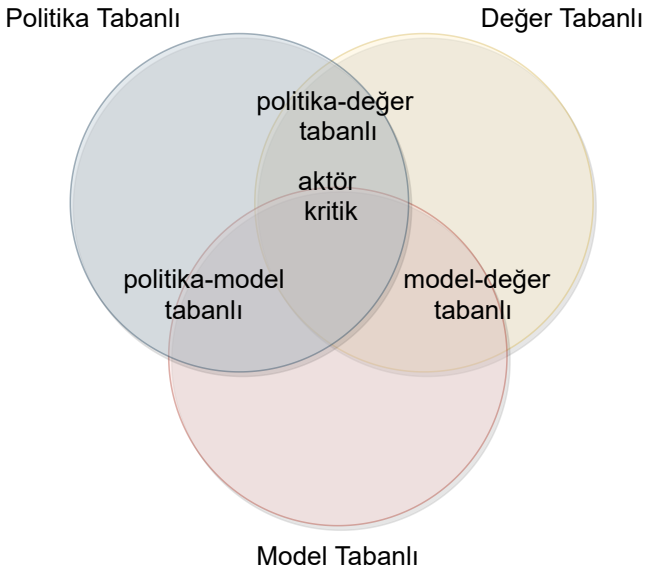
$$V(s) = \max_{a}(R(s,a) + \gamma \sum_{s^{'}} P(s,a,s^{'})V(s^{'})) \tag{4.2}$$

Yukarıdaki denklemlerde $V(s)$ başlangıçtaki durum değerini, $R(s,a)$ bir sonraki durum için ödülü, $V(s^{'})$ sonraki durumun değerini, $\max_{a}$ aksiyon uzayındaki maksimum aksiyon değerini, $P(s,a,s^{'})$ a eylemi yapıldığında bitme olasılığını ve γ ise indirim oranını

ifade eder. Pekiştirmeli öğrenmede Bellman denklemlerini çözmek için değer yinelemeli (aktör) veya politika yinemeleli (kritik) algoritmaları kullanılır. Bu algoritmalara Bölüm 4.4'de kısaca değinilmiştir.

4.4 Pekiştirmeli Öğrenme Algoritmaları

Pekiştirmeli öğrenme yöntemlerinin uygulanmasına göre farklı yaklaşımlar bulunmaktadır. Bu yaklaşımlar öğrenme hedeflerine göre politika tabanlı ve değer tabanlı, çözüm yöntemlerine göre model tabanlı ve modelden bağımsız olarak adlandırılır. Pekiştirmeli öğrenmenin her şeyi kapsayan bir algoritma taksonomisini çizmek birbiri içine geçmiş uygulama yöntemleri nedeniyle oldukça zordur. Bu yüzden kapsam sınırları Şekil 4.2 küme yöntemi ile gösterilmeye çalışılmıştır.



Şekil 4.2: Pekiştirmeli öğrenme yöntemleri (Ünlü, 2019)

4.4.1 Politika Tabanlı Öğrenme (Sadece Aktör)

Politika tabanlı öğrenme yöntemlerinde amaç maksimum ödül kazanılmasını sağlayacak en iyi politikayı bulmaktır. Bu yöntemler, belirli bir durum için olma

olasılığını, kontrol politikasının belirlendiği şekilde tahmin eder (Bhagat & Banerjee, 2018). Politika tabanlı yaklaşımların avantajı, yeterince küçük bir öğrenme oranı verildiğinde ilgili miktarı doğrudan optimize etmeleridir. En büyük dezavantajları ise örnek verimsizliğidir. Yani politika gradyanları, dağıtımlardan hesaplandığı için, varyans genellikle aşırıdır. Politika güncellemeleri uygun geometri kullanılarak geliştirilse bile varyans azaltma ihtiyacı her şeyden önemlidir (Nachum, Norouzi, Xu, & Schuurmans, 2017).

4.4.2 Değer Tabanlı Öğrenme (Sadece Kritik)

Değer tabanlı pekiştirmeli öğrenme algoritmaları, değer işlevlerini veya eylem değeri işlevlerini optimize etme esasına dayanır. Bu işlevler, durumlara veya durum-eylem çiftlerine ilişkin belirli bir politikanın beklenen ödülleridir. Değer tabanlı öğrenme de özyinelemeli döngüler kullanarak optimal durum-eylem değeri fonksiyonuna ulaşılmaya çalışılır. En iyi durum-eylem değerini bulma işlemi, önceki durumlardan ve eylemlerden bağımsız olarak yürütülür.

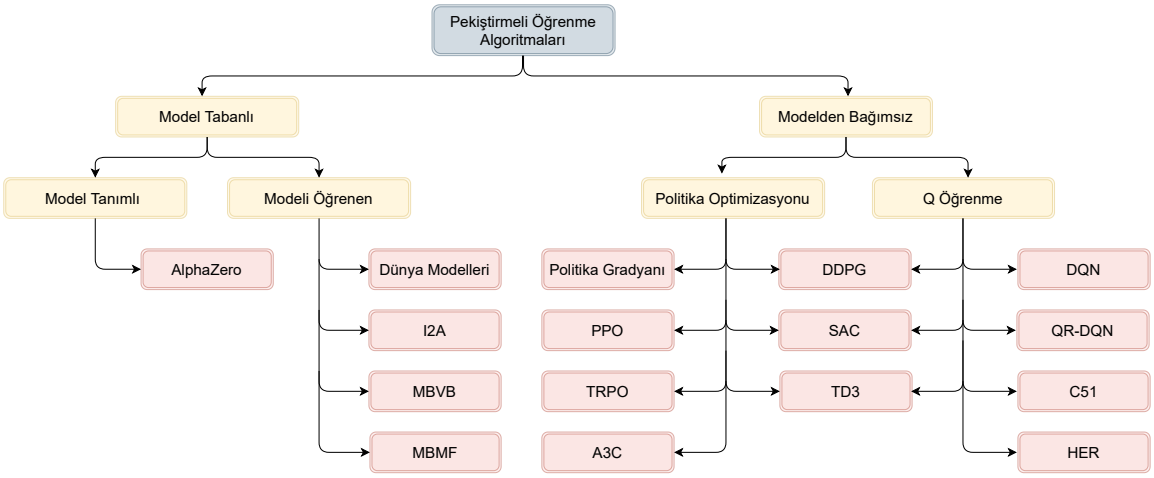
4.4.3 Modele Göre Öğrenme

Modele göre öğrenme Model tabanlı ve Modelden bağımsız olarak ikiye ayrılır.
Model tabanlı pekiştirmeli öğrenmede, ortamdaki geçişlerin ve anlık sonuçların dahili bir modelini oluşturmak için deneyimi kullanır. Daha sonra bu dünya modelinde arama veya planlama yoluyla uygun eylemler seçilir (Dayan & Niv, 2008).

Modelden bağımsız pekiştirmeli öğrenmede ise, optimum davranışı elde edebilmek için, bir dünya modelini tahmin etmeden veya kullanmadan iki basit nicelikten (durum/eylem değerleri veya politikalar) birini veya her ikisini doğrudan öğrenmek için deneyim kullanılır (Dayan & Niv, 2008).

Modelden bağımsız yöntemler istatistiksel olarak model tabanlı yöntemlere göre daha

az etkilidir. Sebebi ise çevreden gelen bilgilerin, doğrudan kullanılmaması, bunun yerine önceki ve muhtemelen hatalı olan tahminler veya durum değerlerinin yeni tahminlerle birleştirilmesidir (Dayan & Niv, 2008). Şekil 4.3'de pekiştirmeli öğrenmenin modele göre ağaç yapısı verilmiştir. Bu çalışmada simülasyondaki akıllı varlıkların eğitilmesi için modelden bağımsız olan proksimal politika optimizayonu (İng., Proximal policy optimization, PPO) algoritması kullanılmıştır.



Şekil 4.3: Modele göre pekiştirmeli öğrenme taksonomisi (Spinningup, 2021)

4.5 Q-Öğrenme

Q-öğrenme, modelden bağımsız ve politika tabanlı olmayan bir RL algoritmasıdır. Q-öğrenme sonucunda matematiksel nihai bir Q fonksiyonu elde edilmeye çalışılır. Bellman Denklemine dayanan Q fonksiyonu Denklem 4.3'de verilmiştir.

$$Q(s,a) \leftarrow r(s,a) + \gamma \underset{a'}{max} Q(s^{'},a^{'}) \tag{4.3}$$

Q-öğrenmede amaç sanal varlığın bulunduğu s anında yapabileceği tüm a eylemleri içinde Q(s,a)'yı maksimize eden eylemi seçmektir. Sanal varlık bulunduğu durumdaki yerel Q değerlerini kullanarak global olarak optimal eylem dizisi bulabilir. Bu şekilde

yaptığı bir eylemin hangi duruma gideceğini bilmese de optimal eylemi seçebilir. Başka bir ifadeyle, bulunulan durum ve eylemin Q değeri, s durumunda a eylemi seçilmesi ile gelecekte kazanılacak azaltılmış toplam ödülü belirlemek için gerekli tüm bilgiyi vermektedir (Mitchell, 1997).

Q-öğrenmede Q(s,a)'yı en iyileyen eylemi seçmek için, Q tablosu kullanılır. Q tablosu, her bir durum-eylem çifti için Q değerleri içerir. Başlangıçta tüm değerleri sıfır olan tablonun her iterasyonda Q değerleri güncellenir.

Aşağıda Q Algoritmasının çalışma prensibi gösterilmiştir (Mitchell, 1997).

- Her s,a için tablodaki Q değerlerine sıfır aktar.
- s durumunu algıla
- Aşağıdaki eylemleri sürekli tekrarla
  - Bir a eylemi seç ve gerçekleştir
  - Anlık r ödülünü al
  - Yeni s' durumunu algıla
  - Q tablosundaki $\hat{Q}(s,a)$ değerini Denklem aşağıdaki gibi güncelle

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \max_{a'} \hat{Q}(s',a')$$

  - $s \leftarrow s'$

4.6 Soft Aktör Kritik

Soft aktör kritik modelden bağımsız ve politika dışı bir pekiştirmeli öğrenme algoritmasıdır. Modelden bağımsız RL algoritmaları, zorlu karar verme ve kontrol görevle-

rinde sıkça kullanılmaktadır. Fakat bu algoritmalar tipik olarak iki büyük problemden muzdariptir. Problemlerden ilki çok yüksek örnek karmaşıklığı ikincisi ise titiz hiperparametre ayarlamasını gerektiren kırılgan yakınsama özellikleridir. Karışılaşılan zorlukların her ikisi de, bu tür yöntemlerin karmaşık, gerçek dünya alanlarına uygulanabilirliğini ciddi şekilde sınırlamaktadır (Haarnoja, Zhou, Abbeel, & Levine, 2018). Bahsedilen problemlere çözüm olmak için geliştirilen ve bu çalışmada da kullanılan algoritmalardan biri soft aktör kritiktir. Soft aktör kritik algoritmasında varlık beklenen ödülü maksimize ederken aynı zamanda entropiyi de maksimize etmeyi hedeflemektedir. Burada entropi olarak adı geçen terim politikada beklenen getiri ve politikadaki rastgeleliğin bir ölçüsüdur. Entropi değerinin artması daha fazla keşif anlamına gelir. Artan keşif öğrenmeyi hızlandırabilir ve politikanın zamanından önce kötü bir yerel optimum değerine yakınsamasını önleyebilir. Bu açıklamalar doğruldusunda algoritmadaki, temel amacı mümkün olduğunca rastgele hareket ederek görevde başarılı olmak olarak tanımlayabiliriz (Haarnoja et al., 2018).

Entropi ile düzenlenmiş RL denklemi, aşağıda Denklem 4.4'de verilmiştir: (Haarnoja et al., 2019).

$$\pi^* = \underset{\pi}{argmax} \sum_t E_{(s_t;a_t\sim\rho_\pi)} \left[R(s_t,a_t) + \alpha H(\pi(\cdot|s_t))\right] \tag{4.4}$$

Burada α parametresi, entropi teriminin ödüle göre görecelik önemini belirler ve böylece optimal politikanın stokastisitesini kontrol eder (Haarnoja et al., 2019). $H(\pi(.|s_t))$, s_t durumundaki politikanın entropisidir ve α, keşif ile sömürü arasındaki ödünleşimi kontrol eder (Wang, Diao, Xu, Shi, & Wang, 2021).

SAC'de kullanılan entropi düzenlemesi için Bellman denklemi, denklem 4.5'deki gibi yeniden oluşturulabilir (Haarnoja et al., 2019):

$$Q^\pi(s,a) = \underset{s^{'}\sim P}{E} [R(s,a,s') + \gamma(Q^\pi(s^{'},a') + \alpha H(\pi(\cdot|s^{'})))] \tag{4.5}$$

4.7 Proksimal Politika Optimizasyonu

Proksimal politika optimizasyonu (İng., proximal policy optimization, PPO) modelden bağımsız ve politikaya uygun bir algoritmadır. PPO-ceza ve PPO-kırpma olmak üzere iki çeşidi vardır. PPO da soft aktor kritikten farklı olarak optimizayon sırasında çevre kısıtlaması getirilir. Lineer bir yaklaşım vardır, bu yaklaşımı tutarlı kılmak için sadece son güncellemeden ortaya çıkan politika fonksiyonunun deneyimleri kullanılır.

Politika Optimizasyon yöntemleri, politika gradyan yöntemleri ve güven bölgesi (İng., trust region) yöntemleri olarak ikiye ayırmak mümkündür.

- **Politika Gradyan Yöntemleri**: İlk yöntem olan politika gradyan yöntemleri, politika gradyanının bir tahmin edicisini hesaplayarak ve bunu stokastik gradyan yükselme algoritmasına yerleştirerek çalışır (Schulman, Wolski, Dhariwal, Radford, & Klimov, 2017). En sık kullanılan gradyan hesaplama formülü Denklem 4.6'da verilmiştir.

$$\hat{g} = \hat{E}_t[\nabla_\theta \log \pi_\theta(a_t|s_t)\hat{A}_t] \tag{4.6}$$

  Burada π_θ bir stokastik politika ve $\hat{A}_t$, t anındaki avantaj fonksiyonunun bir kestirimidir. $\hat{E}_t$ ise örnekleme ve optimizasyon arasında değişen bir algoritmada, sonlu bir örnek grubu üzerindeki ampirik ortalamayı gösterir. Gradyan kestirimi $\hat{g}$ aşağıdaki 4.7 hedef fonksiyonunun türevi alınarak elde edilir (Schulman, Wolski, et al., 2017).

$$L^{PG}(\theta) = \hat{E}_t[\log \pi_\theta(a_t|s_t)\hat{A}_t] \tag{4.7}$$

  Aynı yörüngeyi kullanarak bu kayıp fonksiyon üzerinde birden fazla optimizasyon adımı gerçekleştirmek genellikle büyük politika güncellemelerine yol açar. Bu du-

rumda, kırpma veya cezanın kullanılmadığı durumla benzer performans gösterir (Schulman, Wolski, et al., 2017).

- **Güven Bölgesi (İng., Trust Region) Yöntemleri**: Politika optimizasyonundaki ikinci yöntem olan güven bölgesi politika optimizasyonunda (İng., Trust Region Policy Optimization, TRPO) (Schulman, Levine, Moritz, Jordan, & Abbeel, 2017) ise hedef fonksiyonu, politika güncellemesinin boyutunda bir kısıtlamaya tabi olarak Denklem 4.8'deki gibi maksimize edilir.

$$\hat{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right] \tag{4.8}$$

Kısıtlama fonksiyonu ise Denklem 4.9'da verilmiştir.

$$\hat{E}_t \left[KL[\pi_{\theta_{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t),] \right] \leq \delta \tag{4.9}$$

Burada θ_{old}, güncellemeden önceki politika parametrelerinin vektörüdür. Hedef fonksiyonunu maksimize etme problemi, hedefe doğrusal bir yaklaşım ve kısıtlamaya ikinci dereceden bir yaklaşım yaptıktan sonra, eşlenik gradyan algoritması kullanılarak yaklaşık olarak çözülebilir. TRPO, aslında bir kısıtlama yerine bir ceza kullanmayı, yani aşağıdaki kısıtlanmamış optimizasyon problemini 4.10 bazı β katsayıları için çözmeyi önermektedir (Schulman, Wolski, et al., 2017). Kullback-Leibler (KL) (Joyce, 2011) sapması politikanın ne kadar değişeceğini belirleyen kısıtlamadır (Heess et al., 2017).

$$\hat{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t - \beta KL[\pi_{\theta_{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t)] \right] \tag{4.10}$$

Mevcut verileri kullanarak bir politika üzerinde mümkün olan en iyi iyileştirmeyi yapma problemini TRPO karmaşık ikinci dereceden yöntemlerle çözmeye çalışır. PPO algoritması ise, TRPO'nun yalnızca birinci dereceden gradyanlara dayanan yaklaşık bir versi-

yonu olarak görülebilir, bu da onu tekrarlayan sinir ağlarıyla (İng., recurrent neural network, RNN) ve geniş ölçekli dağıtılmış bir ortamda kullanımını daha uygun hale getirir (Heess et al., 2017). İki farklı çözüm yöntemi ile ilerleyen PPO'nun çalışma prensibi de bu yöntemlere göre farklılıklar gösterir (Joyce, 2011).

4.7.1 PPO-Ceza

PPO-Ceza algoritması, TRPO gibi KL sapması ile kısıtlanmış politika güncellemesini yaklaşık olarak çözer. Ancak hedef işlevindeki KL sapmasını kısıtlama değil ceza olarak kullanır ve eğitim süresince ceza katsayısını otomatik olarak ayarlar. Her politika güncellemesinde aşağıdaki adımlar gerçekleştirilir (Schulman, Wolski, et al., 2017).

- KL sapmasının ceza olarak kullanıldığı aşağıdaki hedef fonksiyon, eğitim örneklerinin tamamının kullanıldığı pek çok SGD periyodu sonucunda elde edilen değerlerle optimize edilir.

$$L^{KLPEN}(\theta) = \hat{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t - \beta KL[\pi_{\theta_{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t)] \right] \tag{4.11}$$

- Denklem 4.11'deki KL sapması teriminin ortalama kestirim değeri d'nin hesaplanmasında Denklem 4.12 kullanılır. Bir sonraki politika güncellemesi için kullanılacak β hesaplanırken d değeri ile d_{hedef} değeri karşılaştırılır. Burada β'nın başlangıç değeri rasgele seçilir ama algoritma hızlı biçimde doğru β değerini hesaplar.

$$d = \hat{E}_t[KL[\pi_{\theta_{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t)]] \tag{4.12}$$

  - Eğer $d < d_{hedef}/1.5, \beta \leftarrow \beta/2$
  - Eğer $d > d_{hedef} \times 1.5, \beta \leftarrow \beta \times 2$

Yukarıdaki adımlara göre, KL sapması d_{hedef} değerinden belirgin biçimde farklı olduğunda politika güncellemesi daha sık gerçekleşir. Fakat, d'nin d_{hedef}den belirgin biçimde farklı olduğu durumlar nadiren oluşur ve bu durumlarda β'nın optimum değeri

hızlı şekilde bulunur. Yukarıdaki 1,5 ve 2 değerleri sezgisel olarak seçilir, algoritma bu değerlere karşı çok hassas değildir (Schulman, Wolski, et al., 2017).

4.7.2 PPO-Kırpma

PPO-kırpma algoritmasında hedef fonksiyonunda, KL sapma terimi veya herhangi bir kısıtlama yoktur. TRPO algoritmasının maksimize ettiği Denklem 4.8, eğer bir kısıtlama kullanmazsa çok aşırı büyük politika güncellemesine neden olur. Bunu önlemek amacıyla, PPO-kırpma algoritmasında maksimize edilirken politikaya yapılan gereğinden büyük (aşağıda Denklem 4.13'te verilen $r_t(\theta)$'yı 1 değerinden uzaklaştıran) güncellenmeleri önlemek için, hedef fonksiyonunun Denklem 4.14 gibi değiştirilmesi önerilmiştir (Schulman, Wolski, et al., 2017).

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \tag{4.13}$$

$$L^{CLIP}(\theta) = \hat{E}_t[min(r_t(\theta))\hat{A}_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t] \tag{4.14}$$

Burada, epsilon bir hiper parametredir. İkinci terim olan $clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t$, L^{CLIP} denklemini, olasılık oranını kırparak değiştirir ve $r_t(\theta)$'nin $[1-\varepsilon, 1+\varepsilon]$ aralığının ötesine geçme isteğini önler. Sonunda, kırpılmış ve kırpılmamış hedef fonksiyon değerlerinin en küçüğü alınır. Böylece elde edilen son hedef değer, kırpılmamış hedef değer üzerinde bir alt sınır oluşturur. Bu yapıda, olasılık oranındaki ($r_t(\theta)$) değişim sadece hedef fonksiyon iyileşecekse göz ardı edilir, hedef fonksiyon kötüleşecek ise kullanılır. TRPO algoritmasının kullandığı (4.9) hedef fonksiyonu denklemi ile POP-Kırpma algoritmasının hedef fonksiyonu Denklemi (4.15), θ_{old} etrafında birinci dereceden aynı denkleme karşılık gelir. Fakat θ, θ_{old}'dan uzaklaştıkça bu iki denklem farklılaşırlar. Olasılık oranı $r_t(\theta)$, $1-\varepsilon$'da kırpılmış ise avantaj (hedef fonksiyonun yönü) pozitif, $1+\varepsilon$'da kırpılmış ise avantaj negatiftir (Schulman, Wolski, et al., 2017). PPO-kırpma algoritmasının çalışma mantığı aşağıda verilmiştir:

Algorithm 1 Proksimal politika optimizasyonu sözde kodu

Giriş: Başlangıç politika parametreleri olan θ_0 ve kırpma eşik değeri ε

for *k=1,2,...* **do**

Önceki π_θ politikasının örneklerini topla

$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ denklemini hesapla

θ_k politikası için herhangi bir avantaj hesaplama algoritması kullanarak $\hat{A}_t$ avantajlarını hesapla

Politika güncellemesini Adam aktivasyon fonksiyonu ile SGD yöntemi kullanarak aşağıdaki gibi hesapla

$\theta_{k+1} = argmax L^{CLIP}_{\theta_k}(\theta) = argmax \underset{\tau\sim\pi_k}{E} \left[\sum_{t=0}^{T}[min(r_t(\theta))\hat{A}_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t]\right]$

Değer fonksiyonunu bulmak için ortalama kare hatasına (İng., mean squared error) regresyon uygula

end

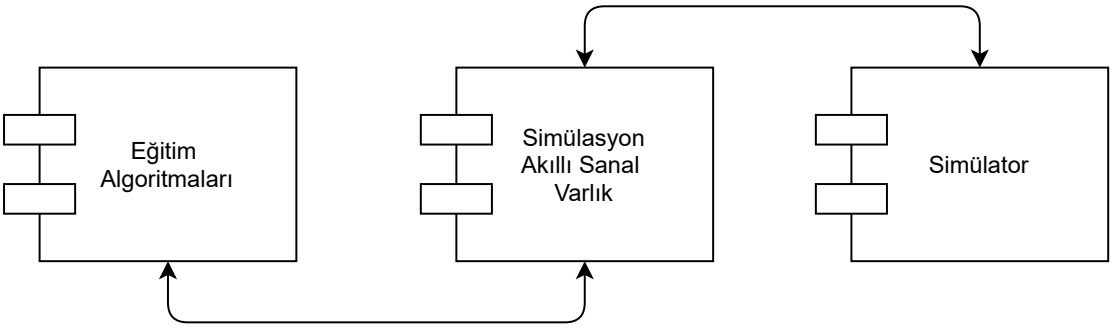
5 DENEY

5.1 Deney Kurgusu

Kullanılan simülasyon üzerinde ilgili problem çözümü için birtakım değişikler gerçekleştirilmiş. Bu doğrultuda deney için oluşturulan tasarım unsurları, proje kapsamında kullanılan donanımsal ve yazılımsal ayrıntılar bu bölümde anlatılmıştır.

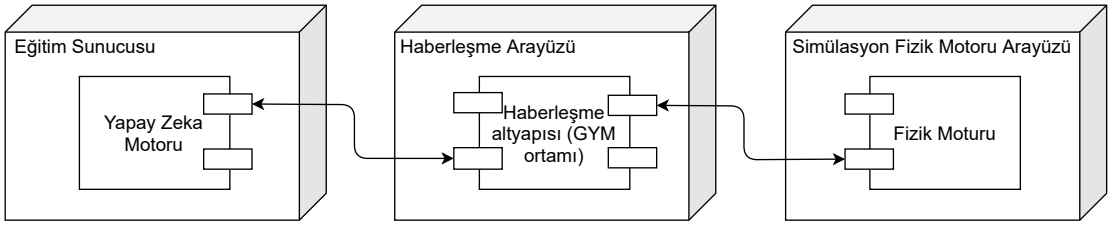
5.1.1 Tasarım Unsurları

Simülasyondaki sanal varlıkların akıllandırılması için eğitim algoritmalarından yararlanılır. Bu algoritmalar, sanal varlıkların optimum davranışları öğrenmesini sağlar. Simülatördeki varlıkların deneyimlerinin yapay zeka fonksiyonlarına dönüştürülebilmesi için eğitim algoritmaları tasarlanmalıdır. Eğitim algoritmaları ile eğitildikten sonra varlıklar, akıllı sanal varlıklara dönüşür. Şekil 5.1'de akıllı varlıklarla donatılmış simülatörlerin tasarlanmasında yer alan üç temel unsur gösterilmiştir. Şekilde görüldüğü gibi taktik çevre simülatörü, içinde barındırdığı akıllı varlıklarla etkileşim içindedir. Akıllı sanal varlıkların deneyimlerinin politika fonksiyonlarına (İng., policy functionlara) dönüştürülebilmesi için simülatörden durum tespiti yapılarak, bu durum ışığında karar verilmesi gerekmektedir. Sanal varlıklara gelen bilgiler ve sanal varlığın aldığı aksiyonlar eğitim algoritmalarına girdi olurlar.



Şekil 5.1: Tasarımda yer alan unsurlar bütünü

Simülasyon tasarımında kullanılan yapı eğitim sunucusu, haberleşme arayüzü ve fizik moturu arayüzü olmak üzere üç alt bölümün birleşmesi ile oluşur. Eğitim sunucusu, içinde yapay zeka moturunu barındıran kısımdır. Haberleşme arayüzü eğitim sunucusu ile fizik moturu arayüzü arasında iletişimi sağlamak için kullanılır. Fizik motoru arayüzü ise simülasyonun fizik kurallarının simüle edildiği bileşenlerinin bulunduğu yazılım bloğudur. Şekil 5.2'de bu ilişkiler anlatılmaktadır.



Şekil 5.2: Projede kullanılan aygıtlar arasındaki iletişim

5.1.2 Donanım ve Ölçeklendirilebilirlik

Projede donanım olarak taktik çevre simülasyonunun yüklendiği bilgisayarlar bulunmaktadır. Ayrıca eğitim işlemlerinin gerçekleştirildiği eğitim sunucusu mevcuttur. Bu sunucu üzerinde 2 tane NVIDIA RTX3090 kartı bulunmaktadır.

Eğitimlerin sunucu makina üzerindeki kullanımlarının analizleri incelendiğinde, eğitim

algoritması 1500 MB hafıza kullanmaktadır ve bu eğitimin karşılık geldiği simülasyonun çalışan kopyaları da yaklaşık olarak 1500 MB kullanmaktadır. Dolayısıyla bir eğitim temel anlamda 3000 MB hafıza kullanmaktadır. 2 RTX kartının toplam hafızasının 48 GB olduğu düşünüldüğünde başka bir kısıtlamanın olmadığı varsayımında maksimum 16 kopya çalışabilir durumdadır.

Eğitimlerin CPU kullanımı incelenecek olursa genel olarak 4 kat hızda çalışan bir simülasyon kopyası %150 ile 200 arasında CPU kullanmaktadır. Eğitim algoritmasının çalıştığı Python kopyası ise %100 CPU kullanmaktadır. Bu açıdan bakıldığında, bir eğitimin ortalamada 3 thread kullandığı varsayılırsa, 32 thread içerisinde maksimum 10-11 tane eğitim çalıştırılabileceği düşünülebilir.

5.1.3 Yazılım

Yazılım kısmı wandb loglama arayüzü ve algoritma şeklinde iki alt başlık altında incelenebilir. Bu alt başlıklardan wandb loglama arayüzünde bir eğitim süresinde gelişimin farklı metriklerle nasıl kayıt altına alındığı gösterilmiştir. Algoritma kısmında ise kullanılan denetimli ve pekiştirmeli öğrenme algoritmaları aktarılmıştır.

5.1.3.1 Performans Metrikleri ve Wandb Kayıt Arayüzü

Wandb (Wandb, 2021) makine öğrenme projeleri için geliştirilmiş eğitim sürecinin takibe alınmasına fayda sağlayan bir kütüphanedir. Wandb sayesinde eğitim sürecinde eğitimin gidişatı hakkında yorum yapmayı mümkün kılan metrikler kayıt altına alınır. Wandb aynı zamanda önceden yapılan eğitimlerin sistematik bir şekilde tutulmasını sağlamaktadır.

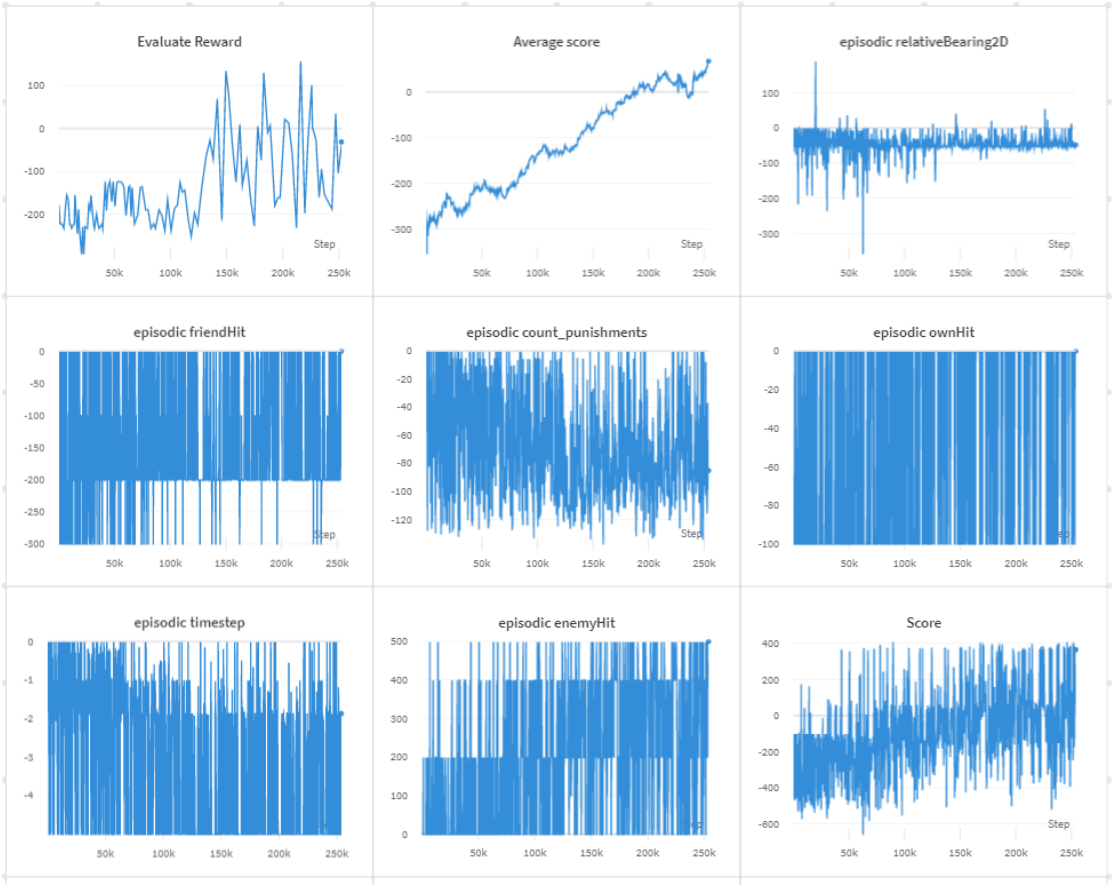
Bu metrikler pekiştirmeli öğrenme için skor (İng., score), ortalama skor (İng., average score), ödül (İng., reward), değerleme ödülü (İng., evalaute reward), kayıp (İng., loss), oyun bölüm sayısı (İng., episode number), iterasyon sayısı(İng., iteration number) ve

her bir ödül kategorisinin değerleri olabilir. Burada ödül anlık olup sistemin her bir gym adımındaki[1] ödülü ifade eder. Skor ise sistemin bir episode, yani başlangıçtan bitişe bir oyun dönemi, sonundaki toplam ödülünü ifade eder. Ortalama skor ise sistemin son 100 oyundaki ortalama skorunu ifade eder. Bu metriğin skor metriğinden en önemli farkı, skor metriğinin öğrenme sırasında keşfetme modundan dolayı belirli bir aralık içinde farklı değerler alabilir olmasıdır. Yani varyasyonun yüksek olma sebebi sanal varlığın çevre içindeki keşfetme maliyetinin skora yansımasından kaynaklanır. Bu maliyeti bir bölüm için görmek daha kolaydır. Bölümlerin ortalaması alınarak ortalama skor hesaplandığında keşfetme ve öğrenilene göre hareket etme maliyetlerini görmek zorlaşır ve varyasyon azalır. Ortalama skor metriği son yüz oyun performansının ortalama bir karakteristiğini ifade ettiği için bu metriğin artması yazılan ödül fonskiyonuna göre öğrenmenin olduğunu ifade eder. Bunun yanında wandb arayüzünden yapılan yumuşatma (İng., smoothing) ile de sistemin ortalama skorunun, skor metriği üzerinden gelişimi gözlemlenebilir. Değerleme ödülü metriğinin aslında ingilizce olarak “evaluate score” ya da ”episodic evaluate reward” olarak ifade edilmesi daha uygun olur. Bu ise eğitim haricinde yapay zeka modelinin keşfetme modunu kapatarak direkt olarak model çıktılarının performansını ölçmeye yarar. Oyun ölüm sayısı eğitimin kaçıncı senaryoda olduğunu ifade eder. İteresyon sayısı ise kaç gym step’i atıldığını ifade eder. Yani diğer bir deyişle kaç kere eylem uygulandığını ifade eder denebilir.

Kullanılan wandb kayıtlarının genel bir görüntüsü Şekil 5.3’de görülmektedir. Burada tüm kayıtlar verilmemiş, bazı metrikler seçilmiştir. Şekil 5.3’de gösterilen hiçbir metrikte yumuşatılma yapılmamıştır. Fakat eğitimin genel performans değerlerini tam olarak ölçmek ve anlamlandırmak için bazı metriklerin yumuşatılmış hallerini incelemek gerekmektedir. Oyun bölümü diye başlayan grafikler ödül ile alakalı olup her bir ödülü ayrı ayrı göstermektedir. Örneğin, bölümdeki düşman vuruşu (İng., episodic enemy hit) adından da anlaşılacağı üzere düşmanı vurma durumunda kazanılan ödül

[1] https://gym.openai.com/

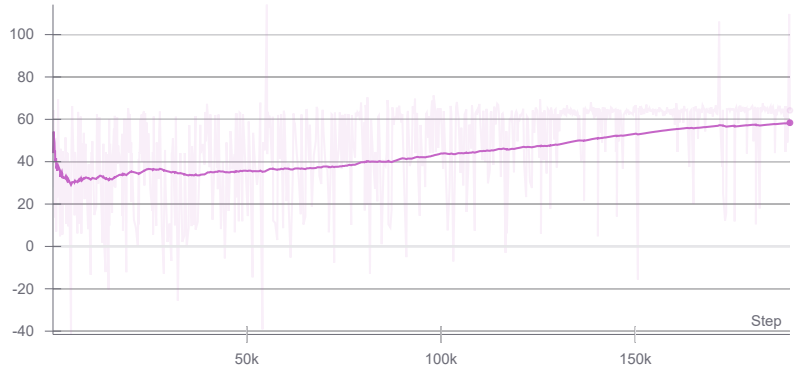
olup zamanla genel anlamda arttığı görülmektedir. Ama artışı burada verilen grafikten anlamak zor olduğu için onun da yumuşatılmış haline bakmak gerekmektedir. Düşmanı vurma ödülü, eğitimleri analiz etmek açısından en önemli metriklerden bir tanesidir. Çünkü amaç son olarak karşı taraftaki muhasımların tamamının yok edilmesidir. Skor ve ortalama skor metrikleri içerisinde diğer ödül birleşenleri de olduğu için veya konsept olarak uygulanabilir olduğu için yumuşatılmış düşman hasarı bilgisi o faktörler olmadan analiz yapmak açısından önem taşımaktadır. Bunun yanında kendini vurma (İng., episodic own hit) bilgisi de dostların kaç kere vurulduğunu analiz etmek açısından bir önem taşımaktadır. Dolayısıyle yumuşatılmış kendini vurma bilgisi de eğitimi analiz etmek için kritik bilgilerden bir tanesidir. Çünkü sonunda en önemli amaçlardan bir tanesi karşı tarafı imha ederken aynı zamanda hayatta kalabilmektir.



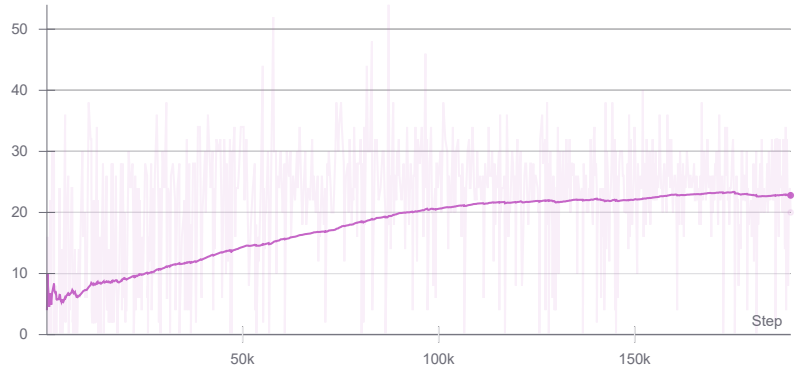
Şekil 5.3: Örnek Wandb kayıt arayüzü

Yukarıdaki ödüllere ek olarak Wandb kayıt altına alma arayüzüne tasarım olarak koyduğumuz birçok ödül bulunmaktadır. Bunlardan bazıları bölüm yönelim ödülü (İng. episodic gimbal reward), bölüm füzeden kaçınma ödülü (İng., episodik missile runaway reward) (Şekil 5.4) ve bölüm düşmana uzaklık ödülüdür (İng., episodic distance to enemy reward). Yönelim ödülü temel olarak düşmana yönelim puanı veya ödülü olarak adlandırılabilir. Bölüm füzeden kaçınma ödülü, varlığın RWR sinyal geldiğinde jammer açma ödülü olarak tanımlanabilir. Bölüm düşmana uzaklık ödülü ise düşmanın uzaklığına bağlı olarak, belirli bir bölgede bulunma durumunda ödül alınan ve düşmana çok yaklaşma veya çok uzaklaşma durumunda ise ceza alınan bir ödül tasarımıdır. Bu-

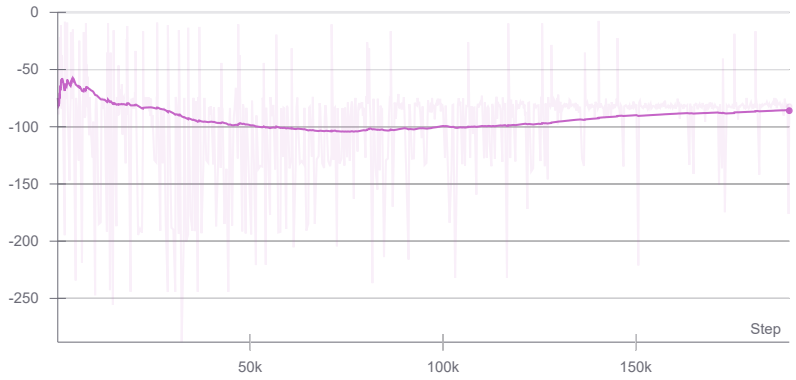
rada bahsi geçen ödüllerin hiperparametreleri senaryodan senaryoya çok değişkenlik gösterdiği için ve optimum parametrelerinin bilinmemesinden dolayı bu ödül mekanizmalarının etkileri düşmanı vurma (İng., enemy hit) ve kendini vurma (İng., own hit) dizaynlarına kıyasla oldukça küçük tutulmaya çalışılmıştır. Aksi takdirde yanlış bir ödül fonksiyonunun baskın olması eğitimi istenilen amaçtan uzaklaştırabilir. Örneğin Şekil 5.4 içinde gösterilen örnekte yönelim ve kaçınma ödülleri artış gösterirken düşmana uzaklık ödülü azalma göstermiştir. Bunun sebebi ana hedefler olan düşman öldürme veya vurulmama ile uyuşmazlığı olabilir. Eğitilen sanal varlığın o ödüle uymamayı tercih ettiği görülmektedir. Ancak yönelimin ve jammer açmanın faydalı olduğunu düşünüp bu ödülleri artıracak hareketler yapmayı öğrendiği gözlemlenmektedir.



(a) Bölüm yönelim ödül örneği



(b) Bölüm kaçınma ödül örneği



(c) Bölüm düşmana uzaklık ödül örneği

Şekil 5.4: Diğer ödül tasarımı örnekleri

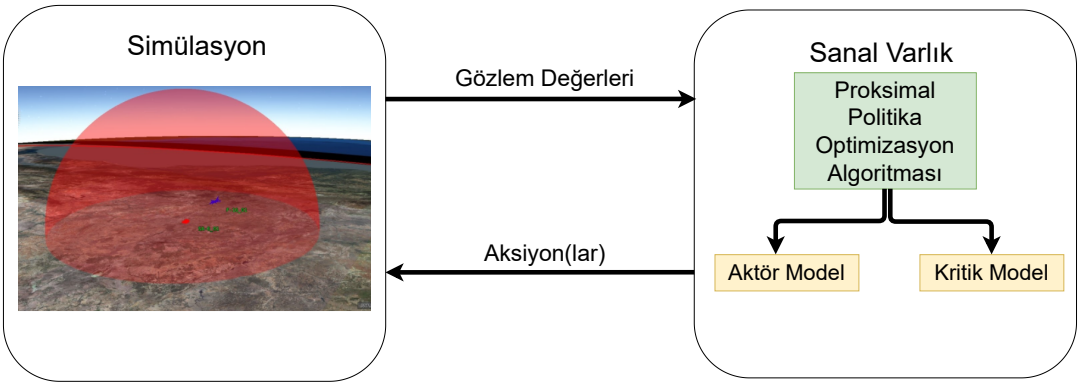
5.1.3.2 Algoritma

Algoritma kısmını temel anlamda denetimli öğrenme ve pekiştirmeli öğrenme olarak ikiye ayırmak gerekir. Denetimli öğrenme kural tabanlı varlıklardan elde edilen tecrübelerin hızlı şekilde makine öğrenmesine aktarılması amacıyla kullanılmaktadır ve bu konuda yapılanlar bu tezin kapsamı dışındadır. Pekiştirmeli öğrenme ise yapay zekanın deneme yanılmayla karşı tarafa üstünlük sağlamasına olanak tanıyacak yeni taktikleri keşfetmesine yarayan öğrenmedir. Bu iki kısmın uyumlu olması denetimli öğrenme ile elde edilmiş yapay zekanın pekiştirmeli öğrenmede kullanılabilmesi açısından önemlidir.

Denetimli öğrenmede eğitim için gerekli veri, kuralın koşturulması ile elde edilir. Sonrasında bu veriyle durum ve aksiyonlar arasındaki ilişkinin yapay zeka modeli tarafından kavranması beklentisiyle kurulan model eğitilir. Bu bakımdan çevrimdışı (İng., offline) bir öğrenme olarak da ifade edilebilir. Denetimli öğrenme verisi ne kadar farklı durumlarla toplanırsa ve ne kadar çok olursa kural davranışı da daha genel öğrenilir. Örneğin, konum genellemelerini yapmak amacıyla denetimli öğrenme eğitimlerinde belirli derece aralıklarla bazı uçakların konumu değiştirilerek farklı başlangıç noktalarına

karşı genellemeler amaçlanmıştır. Yani varlıkı her defasında farklı bir dairesel konumdan doğarak adım adım bir daire yörüngesini tamamlamaktadır. Bu sayede bu daire içerisinde daha iyi bir genelleme yapabilmektedir.

Pekiştirmeli öğrenme kısmında temel olarak PPO (Schulman, Wolski, et al., 2017) olarak ifade edilen algoritma kullanılmıştır. Bu algoritma aktör ve kritik adı verilen yapılardan oluşmaktadır. Algoritmadaki aktör ve kritik yapı birleşenleri literatürde yetkinliği ispatlamış birçok algoritma bünyesinde kullanılmaktadır (Mnih et al., 2016; Haarnoja et al., 2018; Lillicrap et al., 2019; Arulkumaran, Deisenroth, Brundage, & Bharath, 2017). Aktör tabanlı eğitim aynı zamanda politika tabanlı eğitim olarak da ifade edilirken, durum uzayı verildiğinde hangi aksiyonlar alınması gerektiğine karar veren derin öğrenme yapısıdır. Kritik ise yalnızca öğrenme sırasında kullanılmakta olup öğrenilen modeli test ederken kullanılmamaktadır. Kritik de bir derin öğrenme yapısı olup eğitim sırasında ödülleri tahmin etmek için kullanılır. Ödülleri tahmin eden bu yapı bazı algoritmalarda Bellman denklemlerini sağlamada, bazı algoritmalarda ise ödül varyansını azaltmada kullanılmaktadır. Kritiğin olmasının diğer bir avantajı ise oyun bölümlerinin sonuna varmadan ara noktalardaki deneyimlerden yapay zekanın güncellenmesine olanak sağlamasıdır. Bu teknik literatürde önyükleme (İng., bootstrapping) olarak bilinmektedir (Arulkumaran et al., 2017). Simülasyon ortamındaki sanal varlığın proksimal politika optimizasyonu algoritması ile aksiyon seçme işlemi Şekil 5.5'de gösterilmiştir.



Şekil 5.5: Simülasyon ve sanal varlıklar arasındaki etkileşim

PPO diğer politika optimizasyonu yöntemlerinde olduğu gibi on-policy çalışan yani derin öğrenme parametrelerini yalnızca bir önceki güncellemelerden sonra alınan deneyimlerle yapan bir algoritmadır. Dolayısıyla algoritma çalışırken deneyimlerin toplandığı hafıza her bir güncellemeden sonra sıfırlanmaktadır. SAC (Haarnoja et al., 2018) ve DDPG (Lillicrap et al., 2019) gibi daha farklı kayıp hesaplarına sahip olan algoritmalar bu deneyimleri sıfırlamadan devam etmektedir. Ek olarak PPO algoritmasında diğer çalışan on-policy aktör-kritik yapılarından farklı olarak yapılan güncellemenin belirli bir sınır içinde kalmasını sağlayan gradyan kırpma (İng., gradient clipping) yaklaşımı uygulanmaktadır. Bu yaklaşım matematiksel olarak güncelenecek yeni politka noktasından aksiyon örneklemeleri olmadığından dolayı bir doğruallaştırma varsayımından gelmektedir. TRPO'nun ilk olarak ortaya atıldığı (Schulman, Levine, et al., 2017) çalışmada, bu kısıtlama KL ıraksama (İng., KL divergence) metriği yapılırken PPO'da bu işlem kayıp fonksiyonunda bulunan kırpma yaklaşımı ile yapılmaktadır.

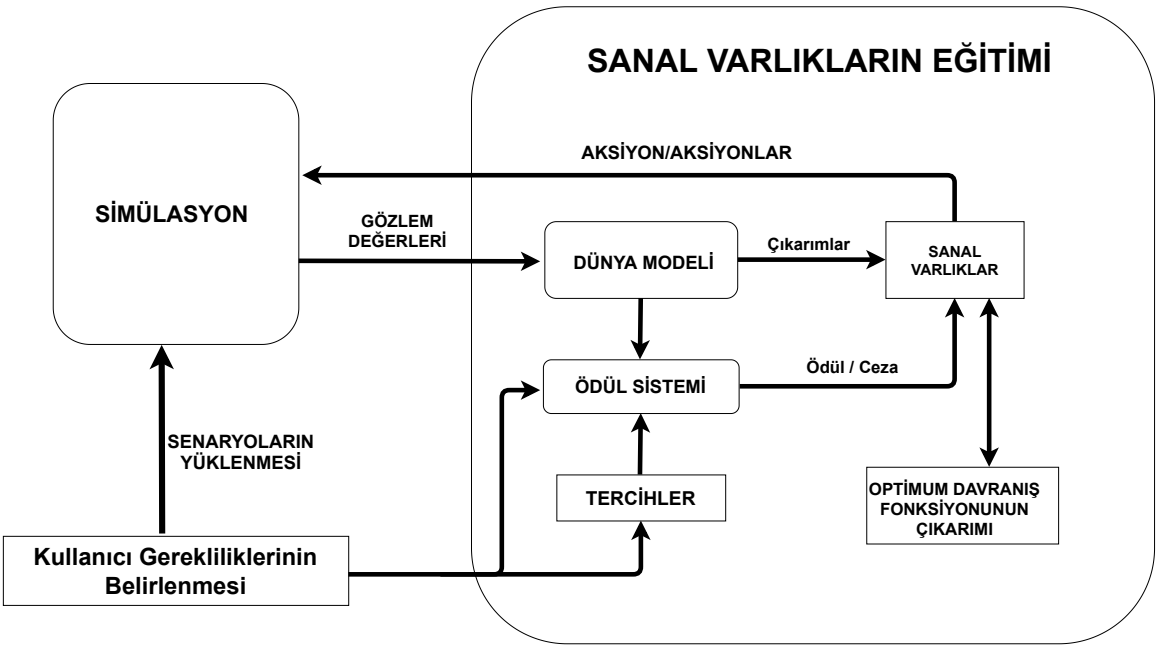
Buna ek olarak literatürde uygulanan RL algoritmaları genel olarak bir gym adımında tek bir aksiyon alınacak şekilde tasarlanmıştır. Örneğin, eğer tanımlanan aksiyon uzayı sağa git, sola git, yukarı git ve aşağı git şeklinde ise, aynı anda hem sağa hem de sola gitmek mümkün değildir. Karmaşık bir askeri senaryoda ise durumlar çok farklıdır. Aynı anda hem manevra yapılması hem de ateş edilmesi gerekebilir. Dolayısıyla tasarlanan

algoritmalarımızda ikili ve dörtlü olmak üzere iki algoritma yapısı tasarlanmıştır. İkili yapıda aksiyon uzayı manevra ve taktik eylemleri olarak ikiye ayrılırken, dörtlü yapıda manevra, öz sensör eylemleri, varlık eylemleri ve mesajlaşma eylemleri olarak dörde ayrılmıştır. Bu ayrım sayısı her bir aksiyon grubundan aynı anda bağımsız aksiyon seçimini mümkün kılmıştır.

Eğitilmiş algoritmaları koştururken temel olarak iki moddan bahsedilebilir. Bunlar argmax modu ve olasılıksal mod olarak ikiye ayrılabilir. Argmax modu bir aksiyonu seçerken her zaman maksimum olasılıklı aksiyonu seçer. Olasılıksal mod ise her zaman en yüksek olasılıklı aksiyonu seçmez, diğer aksiyonları da seçebilir. Bu seçimler her aksiyonun olasılığı ile doğru orantılıdır. Bunu bir nevi hileli bir zar atmaya benzetilebilir. Hileli zarda eğer 6 daha çok olasılık almış ise zarı attığımda daha çok frekansta gelir. Bu kavram pekiştirmeli öğrenmede ortamı keşfe yardımcı olur. Algoritma doğrudan yanlış bir lokal minimaya yönelmek yerine keşfedilmemiş aksiyonlara da olasılık atayarak daha iyi aksiyonları bulmaya çalışır. Denetimli öğrenme algoritmaları eğitim yapıları itibariyle argmax modda çalıştırılmalıdır. Pekiştirmeli öğrenmenin artırmaya çalıştığı ortalama skor olasılıksal modun performansını ifade eder. Değerleme ödülü ise eğitilen pekiştirmeli öğrenme algoritmasının argmax skorunu ifade eder. Pekiştirmeli öğrenme algoritmalarında genel anlamda yumuşatılmış skor metriğinin yumuşatılmış değerleme ödülü metriğine göre daha iyi performans gösterdiğini söylenilebilir.

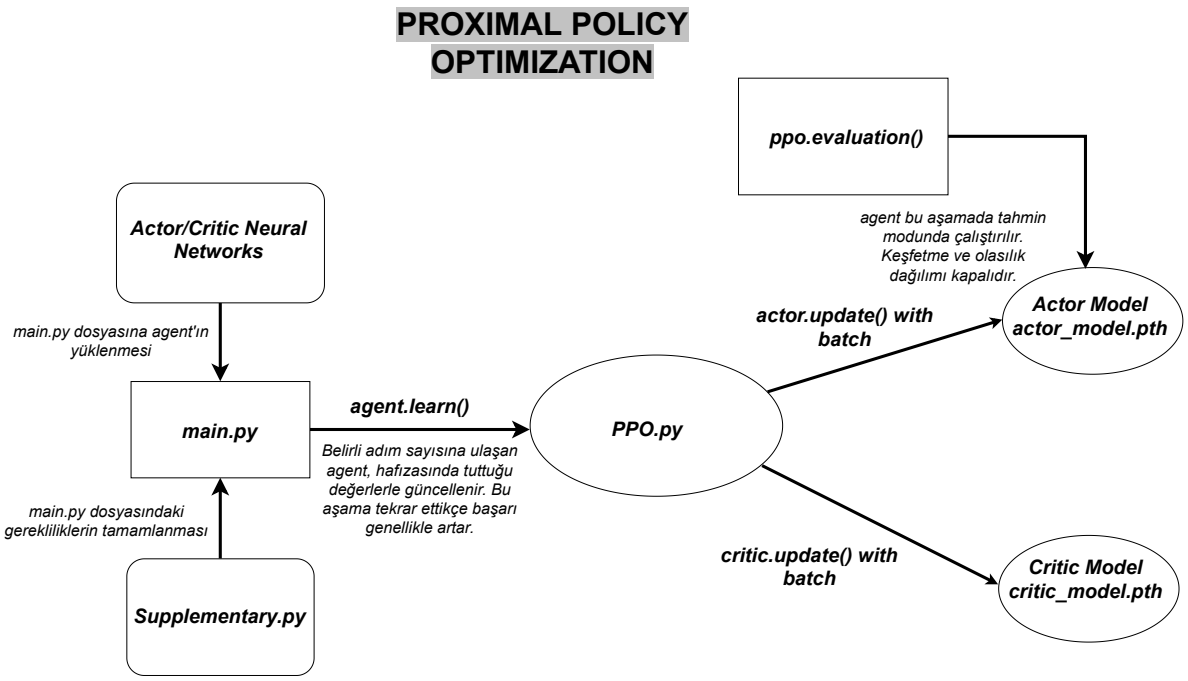
Simülasyondaki sanal varlıkların eğitimini daha ayrıntılı bir biçimde Şekil 5.6'da görebiliriz. Burada öncelikli olarak kullanıcı isterleri belirlenir bu doğrultuda simülasyona gerekli senaryo yüklenir. Ayrıca sanal varlıkların eğitilmesi için gerekli olan ödül ve ceza sistemi de yine bu isterlere göre düzenlenir. Örneğin simülasyon eğitiminde asıl istenen hedef, düşman savunmasını imha etmek ise en çok ödül bu duruma verilmeli iken kullanılan uçağın vurulması sonucunda görevin iptal olması en büyük ceza olarak kabul edilebilir. Gerekli ödül sistemi belirlendikten sonra sanal varlıklar simülasyon

içinde aksiyonlar gerçekleştirirler ve bunun sonucunda simülasyon tarafından gözlem değerleri elde ederler. Simülasyon tarafından nasıl bir karşılık alınacağı dünya modellenmesi kullanılarak da tahmin edilebilir. Bütün bu sistemde asıl erişilmek istenen simülasyon içindeki sanal varlıklar için optimum davranış fonksiyonununa ulaşmaktır.



Şekil 5.6: Sanal varlıkların eğitilmesinin ayrıntılı gösterimi

Şekil 5.7'de Python dili kullanılarak oluşturulan çalışma içinde proksimal politika optimizasyon algoritmasının nasıl kullanıldığı gösterilmiştir.



Şekil 5.7: Python dilinde proksimal politika optimizasyonunun kullanımı

5.1.3.3 Temel Eylemler

Bu çalışmada, taktik çevre simülatöründe kullanılabilecek eylemlerden bazı önemli eylemler seçilerek, aksiyon uzayı kısıtlanmıştır. Aksiyon uzayının çerçevesinin belirlenmesi, algoritma karmaşıklığının düşük tutulması açısından önemlidir. Tablo 5.1'de ve Tablo 5.2'de temel eylem kategorileri listelenmektedir. Onbir adet temel eylem kategorisi bulunmaktadır. Tablo 5.1'da temel eylemler listelenmektedir.

	Temel Eylem Kategorileri
1	Manevra Eylemleri (Parametre detayları Tablo 2'de verilmiştir.)
2	Silah Eylemleri (Silah Tipi, Atış Modu)
3	IR Flare Eylemleri (Flare Sayısı)
4	Chaff Eylemleri (Chaff Sayısı)
5	IR Karıştırıcı Eylemleri (Aç, Kapa, Oto)
6	Radar Karıştırıcı Eylemleri (Aç, Kapa, Oto)
7	Lazer Karıştırıcı Eylemleri (Aç, Kapa, Oto)
8	IFF Eylemleri (Mod, Kod)
9	Duman Eylemleri (Aç, Kapa, Oto)
10	Sensör Eylemleri (Mod, Aç, Kapa)
11	İletişim Eylemleri (Mesaj Gönder, Mesaj Al)

Tablo 5.1: Temel eylem kategorileri

Temel Eylemler	**Parametreler**
Takip et	Takip edilecek platform kimlik bilgisi
	Takip minimum / maksimum mesafesi (metre)
	Takip açısı (derece)
	Takip irtifası (metre)
Konuma git	Koordinat bilgisi
	Beklenen ulaşma zamanı veya ilerleme hızı (knots)
	Dönüş hızı (opsiyonel)(derece/sn)
	Tırmanış/ Dalış Hızı (opsiyonel)
	Hızlanma/yavaşlama ivmesi (opsiyonel) (m/s^2))
	İrtifa (MSL/AGL) (opsiyonel) (metre)
Oyuncuya git	Oyuncu kimlik bilgisi
	Yaklaşma mesafesi (metre)
	Yaklaşma açısı (derece)
	Yaklaşma irtifası (metre)
	Dönüş hızı (opsiyonel)
	Tırmanış/dalış hızı (opsiyonel)
	Hızlanma/yavaşlama ivmesi (opsiyonel) (m/s^2)
İlerle (derece)	Yön bilgisi (derece)
	Hız bilgisi (knots)
	Dönüş hızı (opsiyonel) (derece/sn)
	Tırmanış/dalış hızı (opsiyonel)
	Hızlanma/yavaşlama ivmesi (opsiyonel) (m/s^2)
	İrtifa bilgisi (metre)
Görev Rotasına gir	Rota kimlik bilgisi
	Rota sapma mesafesi (opsiyonel) (metre)
	Rota sapma açısı (opsiyonel) (derece)
	Rota takip türü (opsiyonel)

Tablo 5.2: Temel eylemler ve parametreleri

5.2 Deney Sonuçları

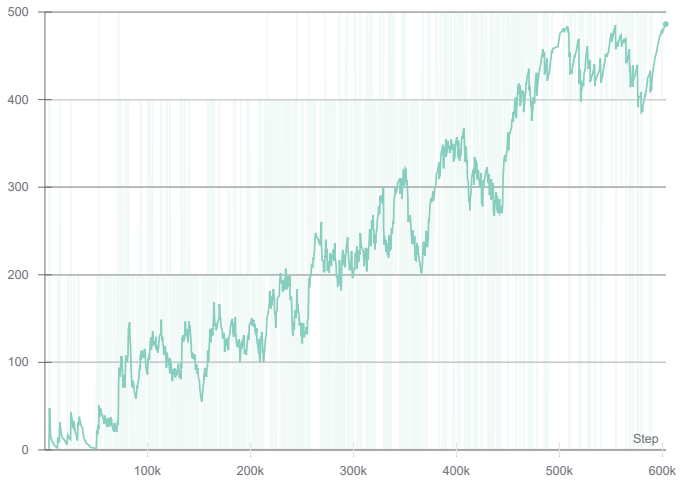
Taktik çevre simülasyonunda bulunan sanal varlıkların akıllandırılması amacına dayanan bu çalışmada kullanılan RL algoritmaları ile başarılı sonuçlar elde edilmiştir. Ulaşılan deneysel sonuçlar temel olarak 3 senaryo üzerinden ele alınmıştır. Bunlar hava hava önleme, sabit kanat hava yer tahrip ve döner kanat hava yer tahrip senaryolarıdır. Bu kısımda eğitimler sadece pekiştirmeli öğrenme ve pekiştirmeli denetimli (İng., supervised RL) öğrenmenin beraber kullanıldığı iki farklı şekilde ele alınmış ve bunlar başlıklarda belirtilmeye çalışılmıştır. Burada wandb arayüzünde kullanılan metrikler üzerinden eğitimlerin analizleri aktarılmıştır.

5.2.1 3e3 Hava-Hava Senaryosu RL Eğitimi

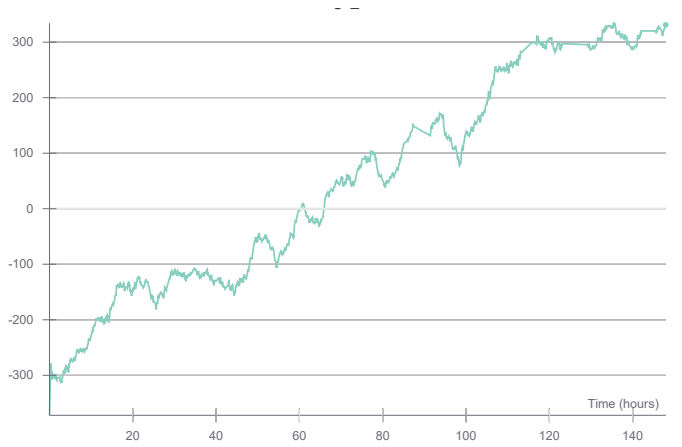
Bu kısımda eğitim için kullanılan senaryo hava hava önleme senaryosudur. Burada üç dost üç düşman ünitesi bulunmaktadır. Eğitim bu dost ünitelerden iki savaş jeti modeli üzerine yapay zeka kuralı eklenerek yapılmıştır. Diğer dost varlık ve düşman varlıklarda kural tabanlı davranış modelleri bulunmaktadır.

3e3 hava-hava senaryosunda eğitilen varlıkların performanslarının zamanla artmasını gözlemek için en tutarlı grafikler düşmanı vurma ödülü ve ortalama skor grafikleridir. Şekil 5.8'te görüldüğü gibi muhasım uçakları vurma konusunda ilk başlarda düşük puan alan varlıklar zamanla onları vurmayı öğrenmektedir. Ödülün en fazla 500 olmasının sebebi ilk iki uçağın vurulmasından 200X2 ödül alan varlıklardan birisi kalan son uçağı vurunca bu bölüm biteceğinden 400+200 şeklinde değil 400+100 şeklinde puan almaktadır. Uçak sayısı ve vurulan uçak başına verilen ödül miktarına göre grafikte farklı sonuçların görülmesi mümkündür. Ortlama skor grafiğinde ise karşı tarafın uçaklarını vurma, onlar tarafından vurulma, onlara doğru yönelme, dosta ateş etme gibi faktörlere göre tasarlanan ödül-ceza mekanizmasından son 100 bölüm boyunca aldıkları toplam ödül miktarı gözlemlenmektedir. Şekil 5.9'daki gibi giderek artmakta olan bir ortalama

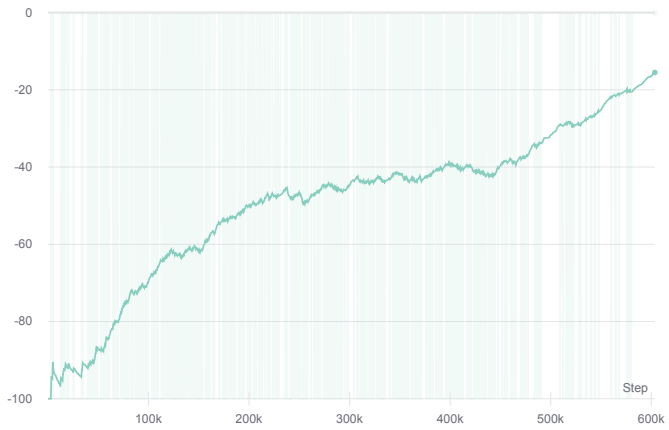
skor grafiği bize varlıkların zamanla öğrendiğini göstermektedir.



Şekil 5.8: 3e3 Hava-Hava senaryosunda eğitilen sanal varlıkların muhasım uçakları vurmaları sonucunda aldıkları ödül miktarının zamana göre değişimi.

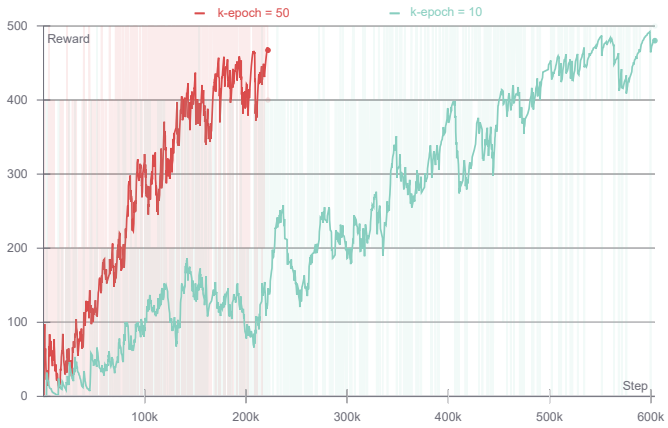


Şekil 5.9: 3e3 Hava-hava senaryosunda eğitilen sanal varlıkların son 100 bölümden aldığı ortalama ödül değerinin zamana göre değişimi.



Şekil 5.10: 3e3 Hava-Hava senaryosunda eğitilen varlıkların kendilerini öldürme durumuna göre aldıkları ödül miktarı. Zaman içerisinde ölümler azalmakta, sanal varlıklar güzel bir öğrenme trendi yakalamaktadır.

Bu aşamada bazı hiperparametre değişiklikleri yapılmıştır. Öğrenmeyi hızlandıran en iyi parametre değişikliği PPO modelinin hafızada tuttuğu veriyle kendini kaç defa güncelleyeceğini ifade eden K-epoch ile sağlanmıştır. Şekil 5.11'de K-epoch değerinin 10'dan 50'ye çıkarıldığında eğitimi hızlandırdığı görülmektedir.



Şekil 5.11: Muhasım uçakları vurma sonucunda elde edilen ödül değerinin K-epoch=10 ve 50 için kıyaslanması. K-epoch artırıldığında, sanal varlıklar muhasım uçakları vurmayı daha hızlı öğrenmiştir.

5.2.2 Hava Yer Tahrip Sabit Kanat RL Eğitimi

Bu kısımda inceleyeceğimiz senaryo sabit kanatlı hava yer tahrip senaryosu olacaktır. Bu senaryoda eğitim 3 düşman 2 dost ünitesi olacak şekilde gerçekleşmiştir. Dost ünitelerinden birinde yapay zeka bulunmaktadır. Diğer varlıklarda kural tabanlı davranış modelleri bulunmaktadır.

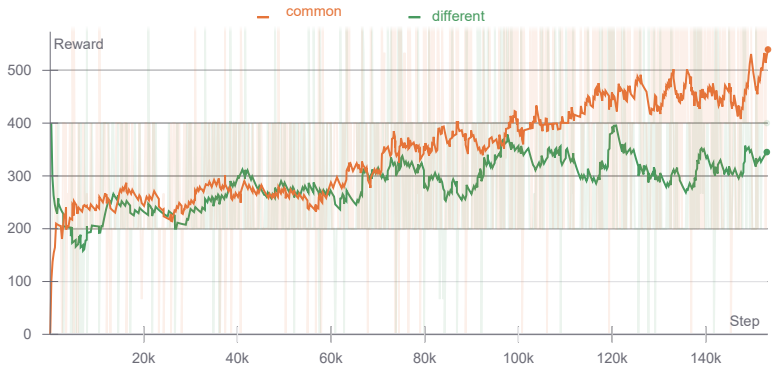
Burada hava-hava senaryosundaki eğitim algoritması yapısal olarak farklılaştırılmıştır. Hava-hava senaryosunda varlığın alabileceği tüm eylemler tek bir model tarafından belirlenirken, bu senaryo eğitiminde eylem kümesi manevra, mesaj, silah ve sensör aksiyonları olarak dörde ayrılmıştır (İng., four worker RL) ve her kümeden aksiyon seçimini farklı bir model yapmaktadır. Bu bölünmenin hata tespiti açısından da katkıları olmuştur. Örneğin manevra ortalama skor grafiğinde öğrenme eğilimi diğer 3 modelin öğrenme eğiliminin başladığı yerden uzun bir süre sonra gözlemlenmeye başlamıştır. Hava-hava senaryosundaki aynı başlıklı grafikler burada da eğitim gidişatı için belirleyici durumdadır. Skorlardaki artışlar eğitimin iyileştiğini yani modellerin öğrendiğini

göstermektedir.

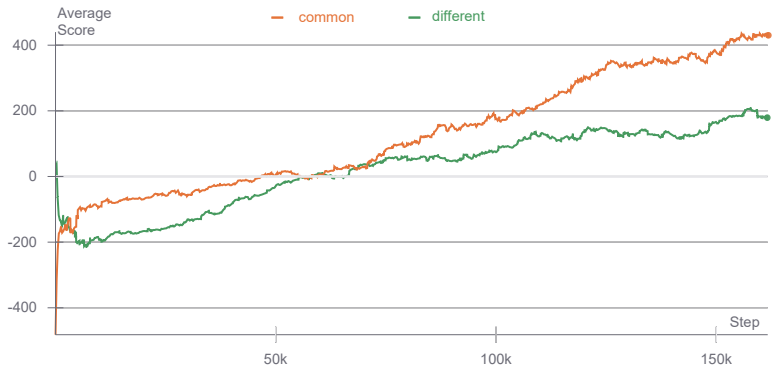
Burada kullanılan 4 model yapısı (İng., four worker) kendi içinde ikiye ayrılmıştır. Modellerin aldığı aksiyonlara topluca aynı ödül ya da cezanın verildiği ortak ödül (İng., common reward) yapısı ile her modelin kendi seçtiği aksiyonlara göre ödül ceza aldığı ikinci yap kullanılmıştır. Kesikli aksiyon kümelerinde seçim yapan modellerde eğitim gayet iyi skorlarla tamamlanırken, manevra aksiyonlarında farklı ödül alan yapı daha başarılı olmuştur. Yani manevra aksiyonlarını seçen modele, kendi ödül ceza sistemini uygulayınca daha güzel manevra hareketleri gözlemlenmiştir. Bu durumun sebebi ortak ödül yaklaşımında manevra ile alakalı ödüllerin seviyesinin diğer ödüllere göre düşük olması olarak gösterilebilir. Manevra modülü farklı ödül tasarımında sadece manevra ödülü ile beslendiğinden manevra ödülünde beklenen hareketleri daha iyi yapabilmektedir. Ama bu da bir yan ödül tasarımı olduğu için, yan ödül tasarımı kötü ise bu durum istenilmeden manevra hareketlerine de sebep olabilir ve düşman vurma hareketlerine de engel olabilir. Dolayısıyle böyle bir tasarımda yan ödüllerin ana ödüle fayda olacak şekilde tasarlanması kritik önem taşımaktadır.

Eğitim sırasında öğrenme olup olmadığı yine düşmanı vurma (Şekil 5.12) ve ortalama skor (Şekil 5.13) grafiklerine bakılarak anlaşılmaktadır.

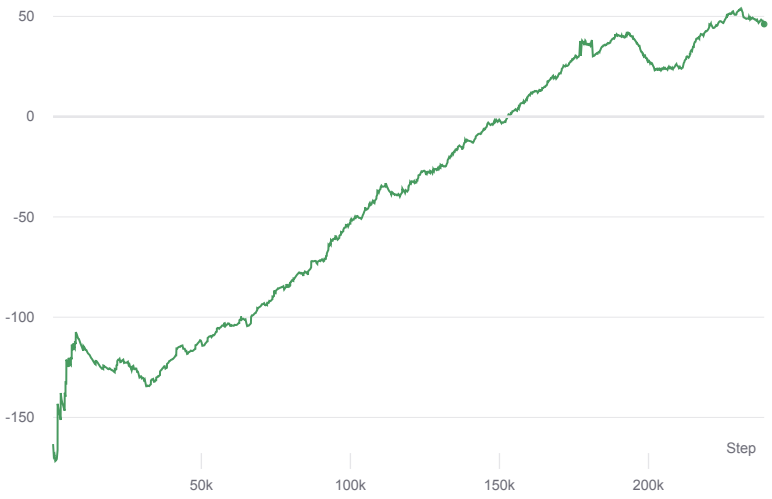
Ortak-ödül ve farklı-ödül yapılarından ortak-ödül yaklaşımında her model aynı merkezde güncellenmektedir. Farklı-ödül yapısında her model kendi ödül yapısı ile güncellendiğinden her modelin ortalama skor grafiği farklı oluşmaktadır. Bu grafikler (Şekil 5.14, 5.15, 5.16 ve 5.17) her aksiyon kümesi için ayrı seçim yapan 4 modelin de farklı öğrenme eğiliminde olduğunu göstermektedir. Manevra modelinde öğrenme diğer modellere göre daha geç başlamaktadır. Manevra aksiyonlarının sonsuz sayıda olmasından dolayı, belirli bir aksiyon dağılım aralığı öğrenmesi beklenen manevra modeli, doğru aksiyonları almaya başlamakta zorlanmaktadır.



Şekil 5.12: Hava yer senaryosunda kurulan 4 aksiyon kümesi için 4 model yapısının ortak-ödül ve farklı-ödül durumunda karşı taraftaki hedeflerini vurabilme grafikleri. Y ekseni düşmanı vurma ödülüne karşılık gelmektedir.



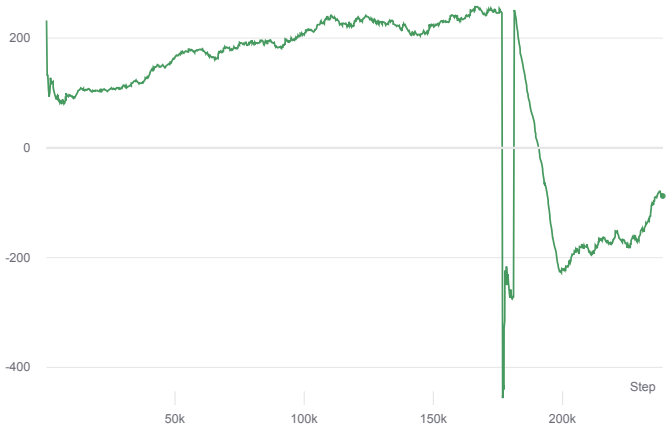
Şekil 5.13: Hava yer senaryosunda kurulan 4 aksiyon kümesi için 4 model yapısının ortak-ödül ve farklı-ödül durumunda tabi tutuldukları hava yer ödül ceza sistemine göre son 100 bölümden aldıkları ortalama skor grafiği.



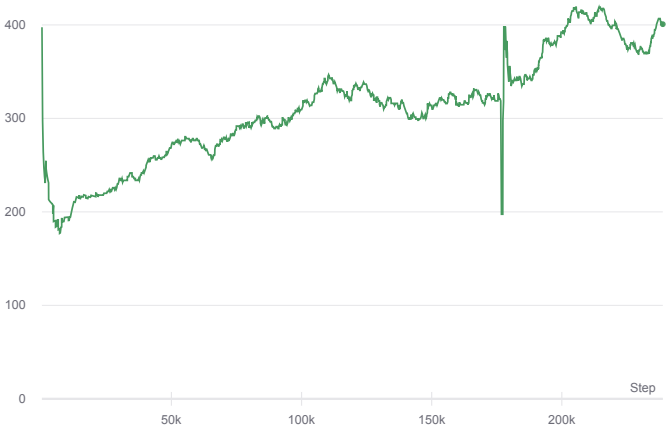
Şekil 5.14: Manevra modelinin ortalama skor grafiği. Biraz geç başlasa da öğrenme eğilimine girdiği gözlemlenmektedir.



Şekil 5.15: Sensör aksiyonlarına karar veren modelin ortalama skor grafiği. Sert iniş çıkışlar eğitimi durdurup farklı değerlerle başlatma denemelerinden kaynaklıdır. Model ana öğrenme eğilimini büyük oranda korumaktadır.



Şekil 5.16: Silah aksiyonlarına karar veren modelin ortalama skor grafiği. Burada silah aksiyonlarında cezada sert bir düşüşün yaşanmasının sebebi eğitimin silah ceza katsayısının artılırılıp devam ettirilmesinden kaynaklanan bir durumdur. Ceza artmış gibi gözükse de aslında toplamda cezanın azalmaya başlandığı gözlemlenmektedir. Dolayısıyla bu bir öğrenme kaybı değildir. Öğrenme sürecindeki silah eylemleri için oluşturulan ödül tasarımının değiştirilmesinden kaynaklanmaktadır.



Şekil 5.17: Mesajlaşma aksiyonlarına karar veren modelin ortalama grafiği. Sert iniş çıkışlar eğitimi durdurup farklı değerlerle başlatma denemelerinden kaynaklıdır. Model ana öğrenme eğilimini büyük oranda korumaktadır.

Ortak-ödül yapısında eğitilen modelin (Şekil 5.12 turuncu renkli ortalama skor grafiğine sahip model) ödül ceza sisteminden, hedefi vurma ödülü, arkadaşını ya da kendini vurma cezası dışındaki tüm ödül cezalar çıkarılmıştır. Burada amaç sistemin sadece ana amaç olan düşmanı imha etme ve hayatta kalmaya tamamıyle odaklanmasını istemektir. Ardından önceki RL eğitiminden en yüksek ortalama skor değerinde kaydedilen modeller ile eğitim tekrar başlatılmıştır. Ortalama skor için yeni grafik Şekil 5.18'deki gibi oluşmaktadır. Burada model performansını artırmaya devam etmiştir ama bir dalgalı öğrenme eğilimi gözlemlenmiştir.



Şekil 5.18: Hedefi vurma ödülü, arkadaşını ya da kendini vurma cezası dışında ödül ceza almadan kaldığı yerden devam ettirilen modelin ortalama skor grafiği. Modelin iniş çıkışlara rağmen öğrenme trendine devam ettiği gözlemlenmiştir.

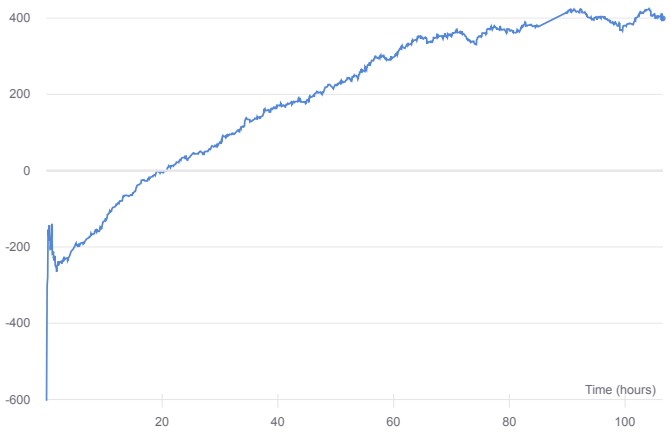
5.2.3 Hava Yer Tahrip Döner Kanat RL Eğitimi

Burada kullanılan senaryo döner kanatlı hava yer tahrip senaryosu olup, eğitim iki dost üç düşman unsur olacak şekilde başlatılmıştır. Bir dost unsurun üzerinde yapay zeka ile yönlendirilen davranış modeli bulunmaktadır.

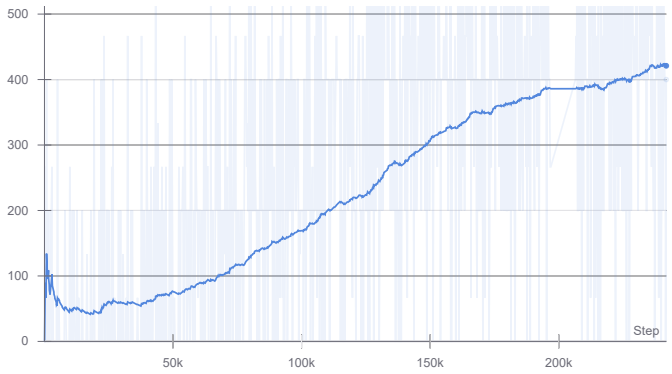
Bu senaryoda tek model (single partition) yapısında eğitim yapılmıştır. Yine ortalama skor (bkz. Şekil 5.19) ve düşmanı vurma (bkz. Şekil 5.20) grafiklerine bakmak

varlıkların öğrenip öğrenemediği konusunda yorum yapmak için yeterli olmaktadır. Bunun yanında başlangıç noktasında rastgele davranışlar alarak müttefiklerini öldüren eğitilen sanal varlığın zamanla bu davranışı da terketmeyi öğrendiği görülmektedir (bkz. Şekil 5.21).

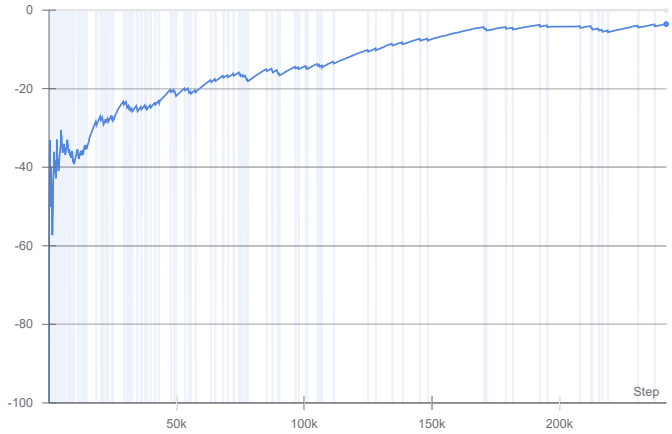
Aynı zamanda bu senaryo özelinde zaman adımı güncelle (İng., update timestep) diye isimlendirilen pekiştirmeli öğrenme algoritmasındaki hiperparametrenin etkisi gözlemlenmiştir. Bu etki Şekil 5.22'de sunulmuştur. Zaman adımı güncelle parametresi, yapay zekanın kaç deneyimden sonra güncelleneceğini belirleyen bir parametredir. Bu sayı şekilde 1000 ve 4000 olarak incelenmiştir. Parametre 1000 olduğunda eğitimin daha hızlı gerçekleştiği gözlemlenmiştir. Fakat eğitimin stabilizasyonu açısından 4000'in daha iyi olduğu gözlemlenmiştir. Bu anlamda eğitimin stabil ilerleyişinin daha önemli olduğu gerekçesiyle bu parametreyi 4000 olarak seçmenin daha uygun olduğuna karar verilmiştir.



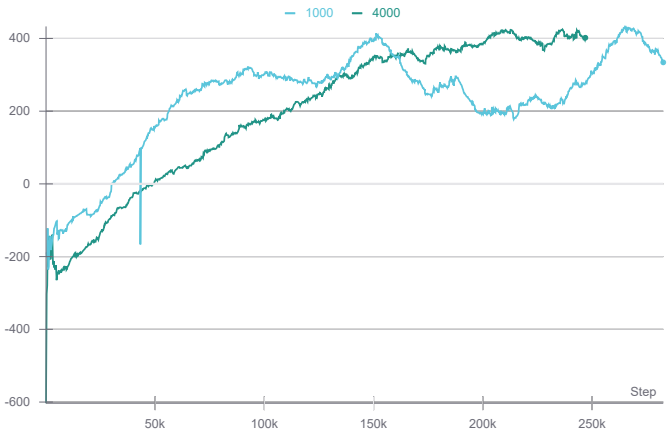
Şekil 5.19: Eğitilen sanal varlığın zamana göre ortalama skor grafiği.



Şekil 5.20: Eğitilen sanal varlığın zamana göre karşı hedefleri vurmadan aldığı ödül grafiği. Sanal varlık, ortalama skor grafiğindeki gibi zamanla daha iyi bir performans göstermeye başlamıştır.



Şekil 5.21: Eğitilen sanal varlığın zamana göre dostunun ölümünün ceza grafiği. Sanal varlık başlangıçta dostuna karşı mühimmat atarken zamanla bu davranışı azaltmayı öğrendiği görülmektedir.



Şekil 5.22: Eğitilen sanal varlıkların son 100 bölümden aldıkları ortalama ödül değerlerinin zaman adımı güncelleme parametrelerinin 1000 ve 4000 için kıyaslanması. Zaman adımı güncellemesi parametesi 1000 olduğunda eğitimin daha hızlı ilerlediği ama ilerleyen aşamalarında stabilizasyon problemleri ile karşılaştığı gözlemlenmektedir.

6 SONUÇ VE DEĞERLENDİRME

Simülatörlerde insan kontrolünün dışında belirlenmiş senaryoya uygun olarak hareket etmesi gereken çok sayıda sanal varlık bulunur. Bu sanal varlıkların tecrübelerden öğrenerek bir zekaya ve adaptasyon yeteneğine sahip olması önemlidir. Bu sayede varlıklar kendilerine verilen rolleri gerçek dünyadakine benzer şekilde gerçekleştirebilecek ve yazılımın amaçlarına daha iyi hizmet etmesini sağlayabilecektir.

Mevcut taktik çevre simülatörü veya herhangi bir oyun yazılımı uygulamalarında kullanıcı tarafından kontrol edilmeyen sanal varlıklar programcılar tarafından hazırlanan kurallar çerçevesinde hareket etmektedirler. Programcı bu varlıkların kontrol edilmesi noktasında karşılaşacağı tüm olasılıkları göz önünde bulundurarak bir algoritma geliştirmek durumunda kalmaktadır. Bu yaklaşım, oluşabilecek senaryoların çeşitliliği ve bu senaryolar sırasında varlıkların alacakları rollerin farklılığı göz önüne alındığında oldukça yoğun emek gerektiren ve her açıdan maliyetli bir iştir.

Sanal varlıkların tezde belirtilen tekniklerle akıllandırılarak taktik çevre simülasyonu gibi yazılımlar içinde kullanılması ile elde edilen sonuçlar bulunmaktadır. Bu sonuçlardan ilki gerçek bir taktik çevre sümülatöründe RL yöntemlerini kullanarak sanal varlıkların akıllandırılabileceğinin gösterilmesidir. Bu bağlamda PPO tekniğinin ilgili problem çözümünde etkili olduğu ispatlanmıştır. Ulaşılan ikinci sonuç simülatörde senaryo oluşturma aşamalarının çok kısalmasıdır. Çünkü her bir sanal varlığın davranış karakteristiklerini içinde bulunduğu senaryoya göre değiştirme ihtiyacı ortadan kalkmıştır. Diğer bir sonuç kural tabanlı yaklaşımda sanal varlıklar aynı durumlarda aynı hataları yapmaya devam ederken, akıllı sanal varlıkların kendilerini

gerçek hayattaki gibi sürekli geliştirerek hatalarından ders çıkarmayı öğrenmiş olmalarıdır. Bunlara ek olarak özellikle eğitim simülatörleri için gerçek hayata yakın bir deneyim ortaya koyulması elde edilen başka bir sonuçtur. Ayrıca eğitilen sanal varlıklar takım olarak davranabilme yeteneği kazanmıştır ve üç farklı senaryo için tasarlanan sistemin kullanılırlığı ispatlanmıştır. Son olarak senaryoya özgü olan sonuçların gösterimi gerçekleştirilmiştir fakat ileriki çalışmalarda deneyim sayısının arttırılması ile senaryodan bağımsız şekilde varlıkların kullanılabileceğinin mümkün olduğu gösterilebilir.

KAYNAKLAR

Armon, T. (2020). *Calculated moves generating air combat behaviour* (Unpublished doctoral dissertation). Leiden University.

Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, *34*(6), 26-38. doi: 10.1109/MSP.2017.2743240

Bagnell, J. A., & Schneider, J. G. (2001). Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings 2001 icra. ieee international conference on robotics and automation (cat. no.01ch37164)* (Vol. 2, p. 1615-1620 vol.2). doi: 10.1109/ROBOT.2001.932842

Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-13*(5), 834-846. doi: 10.1109/TSMC.1983.6313077

Bellman, R. (1957). A Markovian Decision Process. *Indiana Univ. Math. J.*, *6*(4), 679–684. doi: 10.1512/iumj.1957.6.56038

Bhagat, S., & Banerjee, H. (2018, 11). *Deep reinforcement learning for soft robotic applications: Brief overview with impending challenges.* doi: 10.20944/preprints201811.0510.v2

Centre, N. A. (2018). *Smart Bandits* (Tech. Rep. No. May). 1059 CM Amsterdam.

Dayan, P., & Niv, Y. (2008). Reinforcement learning: The Good, The Bad and The

Ugly. *Curr. Opin. Neurobiol.*, *18*(2), 185–196. doi: 10.1016/j.conb.2008.08.003

Fayjie, A. R., Hossain, S., Oualid, D., & Lee, D. (2018). Driverless car: Autonomous driving using deep reinforcement learning in urban environment. In *2018 15th international conference on ubiquitous robots (ur)* (p. 896-901). doi: 10.1109/URAI.2018.8441797

Haarnoja, T., Ha, S., Zhou, A., Tan, J., Tucker, G., & Levine, S. (2019). *Learning to walk via deep reinforcement learning.*

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, *abs/1801.01290*.

Heess, N., TB, D., Sriram, S., Lemmon, J., Merel, J., Wayne, G., … Silver, D. (2017). *Emergence of locomotion behaviours in rich environments.*

Joyce, J. M. (2011). Kullback-leibler divergence. In M. Lovric (Ed.), *International encyclopedia of statistical science* (pp. 720–722). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-04898-2_327

Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, *32*(11), 1238–1274.

Källström, J., & Heintz, F. (2019, 10). Multi-agent multi-objective deep reinforcement learning for efficient and effective pilot training. In (p. 101-111). doi: 10.3384/ecp19162011

Källström, J., & Heintz, F. (2020). Agent coordination in air combat simulation using multi-agent deep reinforcement learning. In *2020 ieee international conference on systems, man, and cybernetics (smc)* (p. 2157-2164). doi: 10.1109/SMC42975.2020.9283492

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., … Wierstra, D. (2019). *Continuous control with deep reinforcement learning.*

Lindberg, J. (2020). *Simulation driven reinforcement improving synthetic enemies in*

flight simulators (basılmamış yüksek lisans tezi).

Mahadevan, S., Marchalleck, N., Das, T. K., & Gosavi, A. (1997). Self-improving factory simulation using continuous-time average-reward reinforcement learning. In *In machine learning-international workshop then conference-* (pp. 202–210).

Michie, D. (1963). Experiments on the Mechanization of Game-Learning Part I. Characterization of the Model and its parameters. *Comput. J.*, *6*(3), 232–236. doi: 10.1093/comjnl/6.3.232

Minsky, M. L. (1954). *Theory of neural- analog reinforcement systems and its application to the brain-model problem (basılmamış doktora tezi).*

Mitchell, T. M. (1997). *Machine learning*. New York: McGraw-Hill.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., ... Kavukcuoglu, K. (2016). *Asynchronous methods for deep reinforcement learning.*

Nachum, O., Norouzi, M., Xu, K., & Schuurmans, D. (2017). Bridging the gap between value and policy based reinforcement learning. *Adv. Neural Inf. Process. Syst.*, *2017-December*(Nips).

Roessingh, J., Rijken, R., Merk, R.-J., Meiland, R., Huibers, P., Lue, T., & Montijn, C. (2011, 12). Modelling cgfs for tactical air-to-air combat training motivation-based behaviour and machine learning in a common architecture..

Schulman, J., Levine, S., Moritz, P., Jordan, M. I., & Abbeel, P. (2017). *Trust region policy optimization.*

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Spinningup. (2021). *A taxonomy of rl algorithms.* https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html. (Erişim tarihi 5/04/2021)

Sutton, R. S., & Barto, A. G. (1999). Book Reviews Reinforcement Learning. *J. Cogn. Neurosci.*, *11*, 126–134.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction.* Camb-

ridge, Massachusetts, Londan, England: MIT press.

Ted, J. (2018). *Tactical simulation in air-to-air combat evolutionary algorithms and behavior tree framework (basılmamış yüksek lisans tezi).*

Teng, T., Tan, A., Ong, W., & Lee, K. (2012). Adaptive cgf for pilots training in air combat simulation. In *2012 15th international conference on information fusion* (p. 2263-2270).

Tesauro, G. (1995). Temporal Difference Learning and TD-Gammon. *Commun. ACM*, *38*(3).

Toubman, A., Roessingh, J. J., Spronck, P., Plaat, A., & Van Den Herik, J. (2015). Rewarding air combat behavior in training simulations. In *2015 ieee international conference on systems, man, and cybernetics* (p. 1397-1402). doi: 10.1109/SMC.2015.248

Wandb. (2021). *Developer tools for machine learning.* https://wandb.ai/site. (Erişim tarihi 15/04/2021)

Wang, S., Diao, R., Xu, C., Shi, D., & Wang, Z. (2021). On multi-event co-calibration of dynamic model parameters using soft actor-critic. *IEEE Transactions on Power Systems*, *36*(1), 521-524. doi: 10.1109/TPWRS.2020.3030164

Wender, S., & Watson, I. (2012). Applying reinforcement learning to small scale combat in the real-time strategy game starcraft:broodwar. In *2012 ieee conference on computational intelligence and games (cig)* (p. 402-408). doi: 10.1109/CIG.2012.6374183

Widrow, B., Gupta, N. K., & Maitra, S. (1973). Punish/Reward: Learning with a Critic in Adaptive Threshold Systems. *IEEE Trans. Syst. Man Cybern.*, *3*(5), 455–465. doi: 10.1109/TSMC.1973.4309272

Wiering, M., Vreeken, J., van Veenen, J., & Koopman, A. (2004). Simulation and optimization of traffic in a city. In *Ieee intelligent vehicles symposium, 2004* (p. 453-458). doi: 10.1109/IVS.2004.1336426

Zuo, Y., Deng, K., Yang, Y., & Huang, T. (2019). Flight attitude simulator control system design based on model-free reinforcement learning method. In *2019 ieee 3rd advanced information management, communicates, electronic and automation control conference (imcec)* (p. 355-361). doi: 10.1109/IMCEC46724.2019.8983893

Ünlü, M. S. (2019). *Teknik analiz ve derin pekiştirmeli Öğrenme ile kriptopara alım-satımı (basılmamış yüksek lisans tezi)* (Vol. 8) (No. 5).

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Fatiha Nur BÜYÜKOFLAZ

EĞİTİM DURUMU

Yüksek Lisans : KTO Karatay Üniversitesi -

Lisans : KTO Karatay Üniversitesi Haziran-2017

İŞ DENEYİMİ

2019	Havelsan	Yazılım Mühendisi
2017	Express Bilgisayar	Yazılım Mühendisi
2016	Sevilog	(Erasmus Stajı) Stajyer Mühendis
2016	Türk Telekom	Stajyer Mühendis

AKADEMİK YAYINLAR

• F. Büyükoflaz ve A. Öztürk, ”Makine Öğrenmesi Algoritmaları ile Çocuklarda Erken Otizm Teşhisi”, 26. IEEE İşaret İşleme ve İletişim Uygulama Konferansı, İzmir, Türkiye, 2018