aws DEV DAY
ONLINE JAPAN

20-22.10.2020

DEV DAY

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

E - 1

Amazon SageMaker RLを活用したUnity環境のAIシミュレーションと自動テスト

保里 善太 Zenta Hori

アマゾン ウェブ サービス ジャパン株式会社
ソリューションアーキテクト

In Partnership with intel

Speaker

Zenta Hori
zenthori@amazon.co.jp

@HoriZenta

保里 善太 (ほり ぜんた)
ソリューションアーキテクト

- 所属

アマゾン ウェブ サービス ジャパン 株式会社
技術統括本部
ゲーム エンターテイメント ソリューション部

- お仕事

AWSをご利用のゲーム企業への技術支援、ゲームのパートナー様支援

- 好きな AWS サービス

AWS Lambda　Amazon SageMaker　Amazon Macie

In Partnership with intel

Agenda

- ゲーム開発と AI の概要について
- Unity と Unity ML-Agents について
- Unity のシミュレーションを Amazon SageMaker RLで実行する
- 実際の QA Automation の流れについて
- デモ
- まとめ

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

ゲーム開発と AI の概要について

In Partnership with intel

ゲーム開発のプロセスについて

ゲームの実装

- ゲームエンジン
  - ゲームエンジンとは、コンピュータゲームのソフトウェアにおいて、共通して用いられる主要な処理を代行し効率化するソフトウェアの総称である。 (Wikipedia)

- 代表的なゲームエンジン
  - Unity
  - Unreal Engine (UE)

ゲームのテスト (Quality Assurance)

- ゲームのテストの意義
  - ソフトウェアバグの発見
  - ゲームバランスの調整・チェック
  - 3Dゲームなどのコリジョン抜けの検出
  - ゲームマップ上の難所の検出

- ゲームバランスとは
  - ゲームバランスは、ゲームを成立させるために取られた均衡（バランス）を総称したもの。特にプレイヤーにとっての難易度と快適さのバランス。ゲームが易し過ぎたり、逆に難し過ぎたりする場合にはゲームバランスは良いとは言えない。 (Wikipedia)

ゲームテストの意義と課題

ゲームテストの意義

- ソフトウェアバグの発見
- ゲームバランスの調整・チェック
- 3Dゲームなどのコリジョン抜けの検出
- ゲームマップ上の難所の検出

ゲームテストの課題

- QAテスターの人件費
- 高い技能や経験がQAテスターに要求される
- 何通りものシナリオをテストするため長時間のテストになる
- 知識伝達やQAテスターの育成の難しさ

ゲームテストへの AI の活用

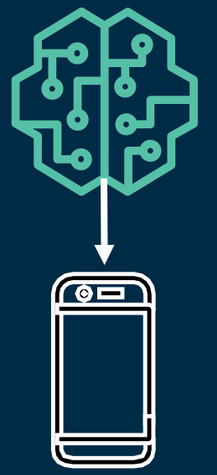
人間によるテスト

Automation

効率化

コスト削減

時間短縮

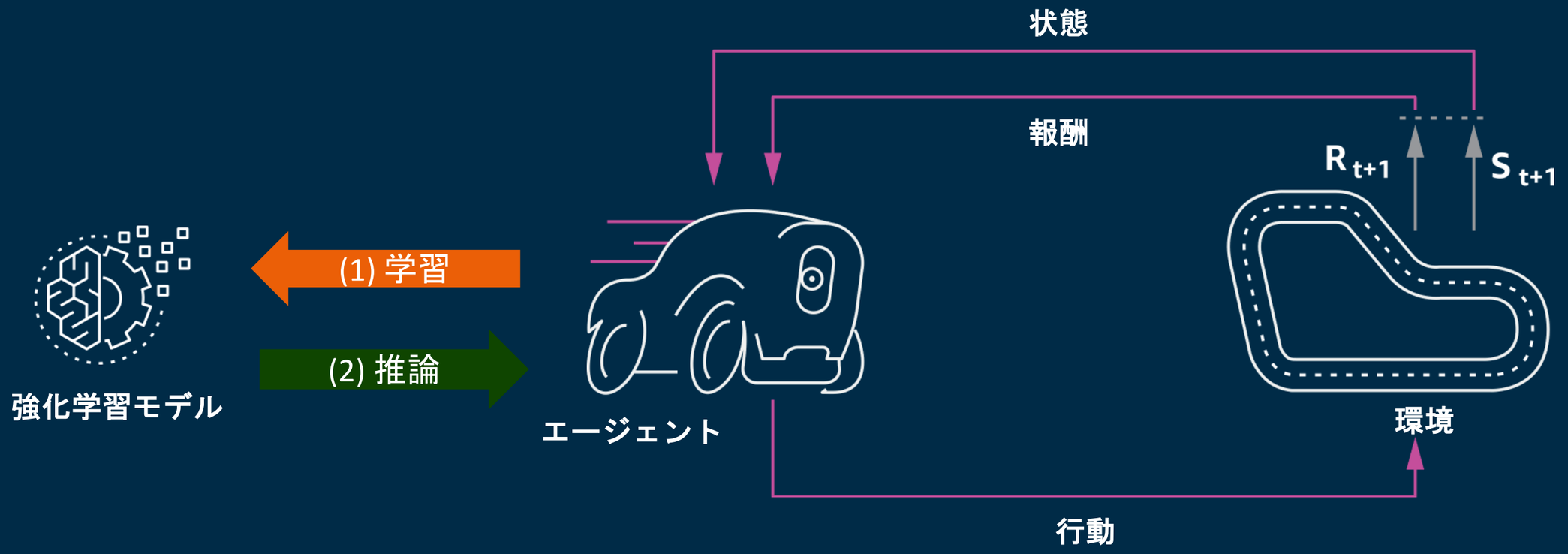


AIによるテスト

- AIによるゲームの自動プレイによりゲームテストの課題を解消
- AIによるゲームの自動プレイには強化学習のアプローチがよく用いられる

In Partnership with intel

AWS DeepRacer を例に強化学習を振り返る



(1) 学習 : エージェントが環境と相互作用しながら、行動に対して報酬を受け取り、ゲームのルールと正しい攻略法を学習していく
(2) 推論 : 学習済みのモデルをエージェントに組み込み、エージェントが自動的に環境内を動き回る

In Partnership with intel

強化学習を用いたゲームテスト自動化のコンセプト

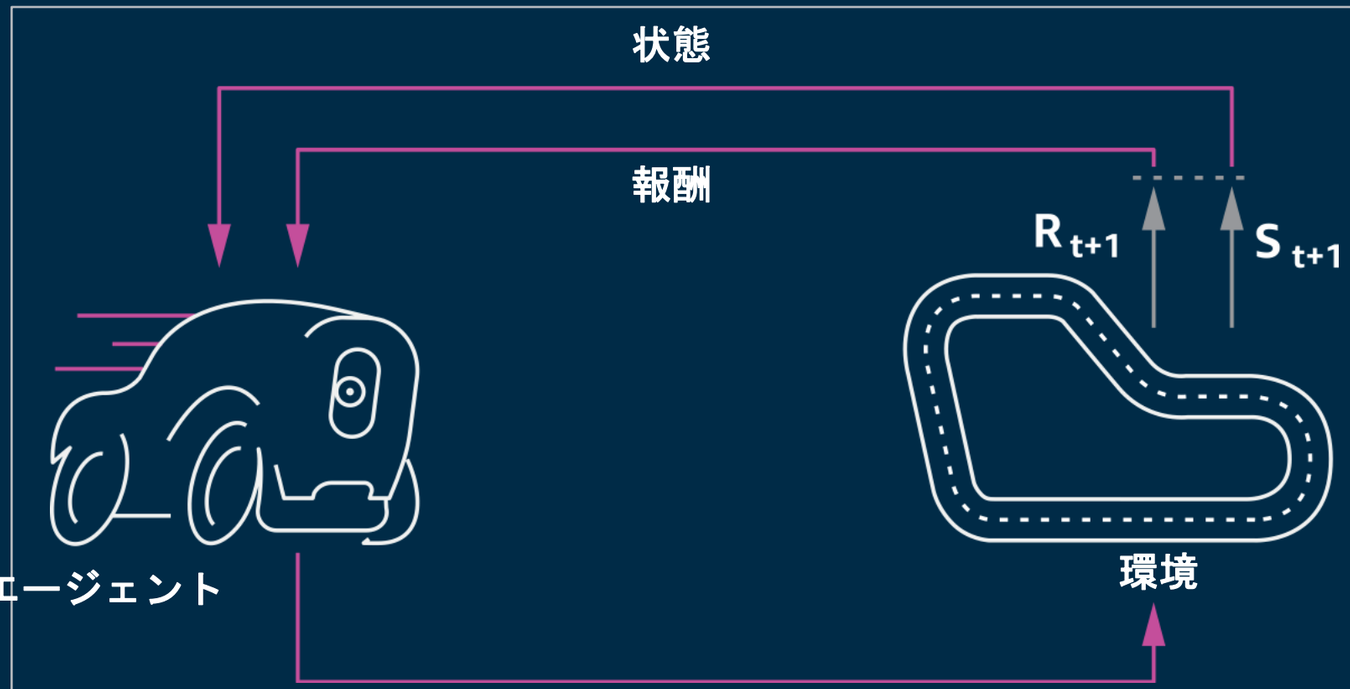
AIを作成してゲーム環境内を自走させる

強化学習モデル

学習

推論

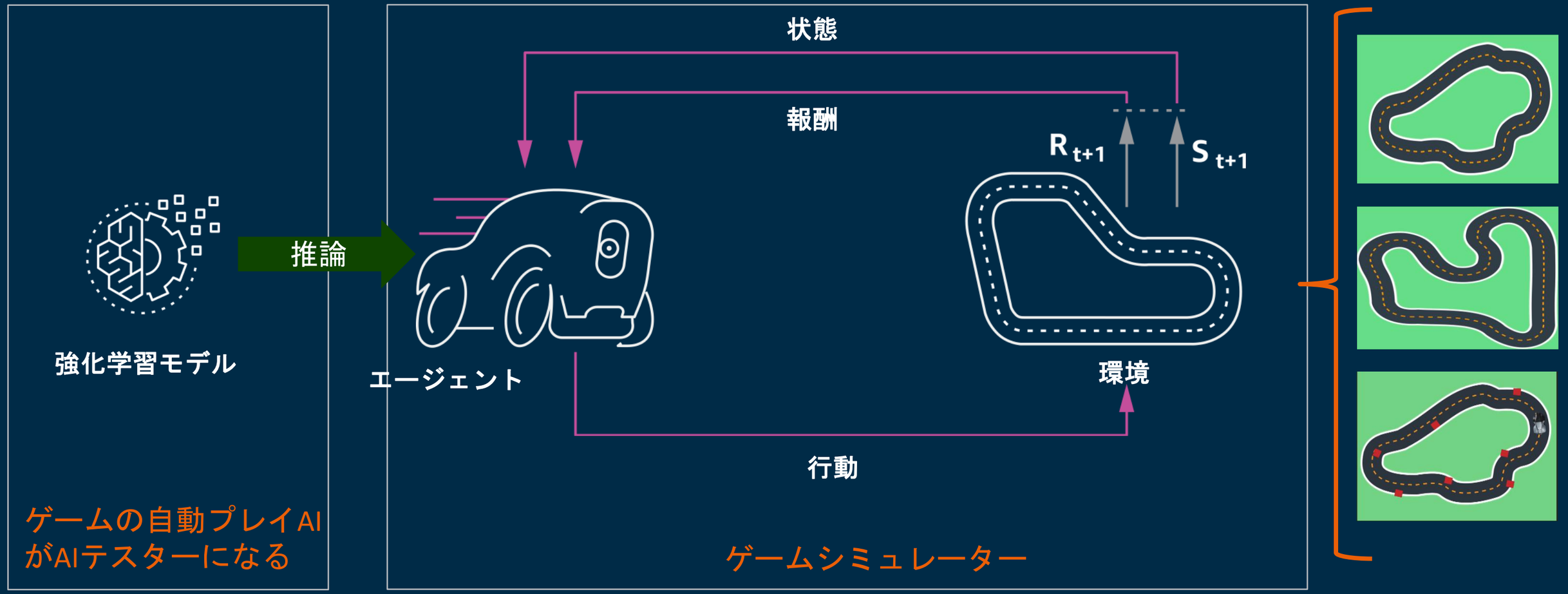
ゲームの自動プレイAIがAIテスターになる



ゲームシミュレーター

In Partnership with intel

強化学習を用いたゲームテスト自動化のコンセプト



新規作成したゲームステージを環境に適用

- ゲームシミュレータ内の環境を新規作成したステージに切り替える
- AIテスターに環境内を自走させて新規ステージの評価を行う

In Partnership with intel

ゲームテスト自動化のステップ

新規ゲーム開発時のテストの自動化への大まかな流れ

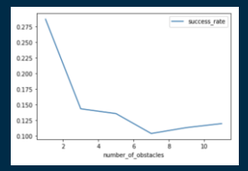
ゲーム制作

強化学習エージェントの学習

モデルのデプロイ

新規ステージ内でエージェントを走らせ評価

ログ収集・結果を可視化



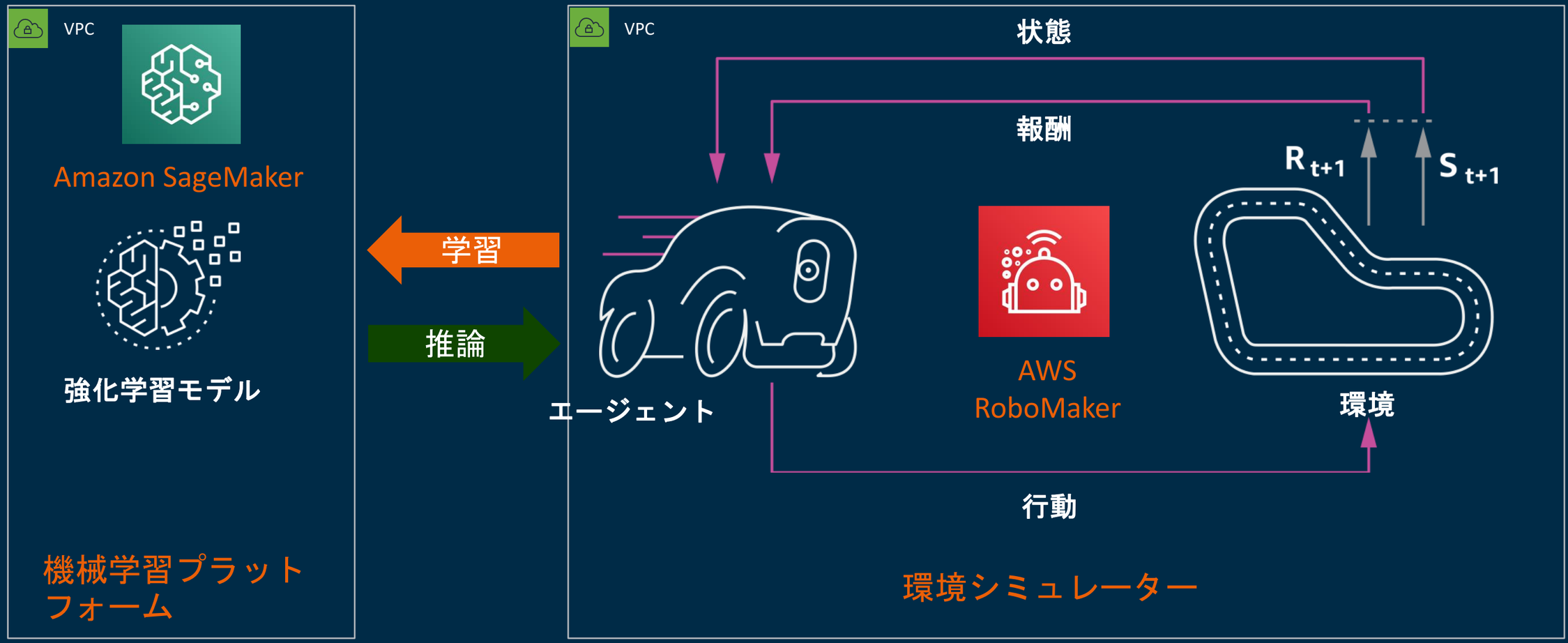
In Partnership with intel

aws DEV DAY
ONLINE JAPAN

Unity と Unity ML-Agents について

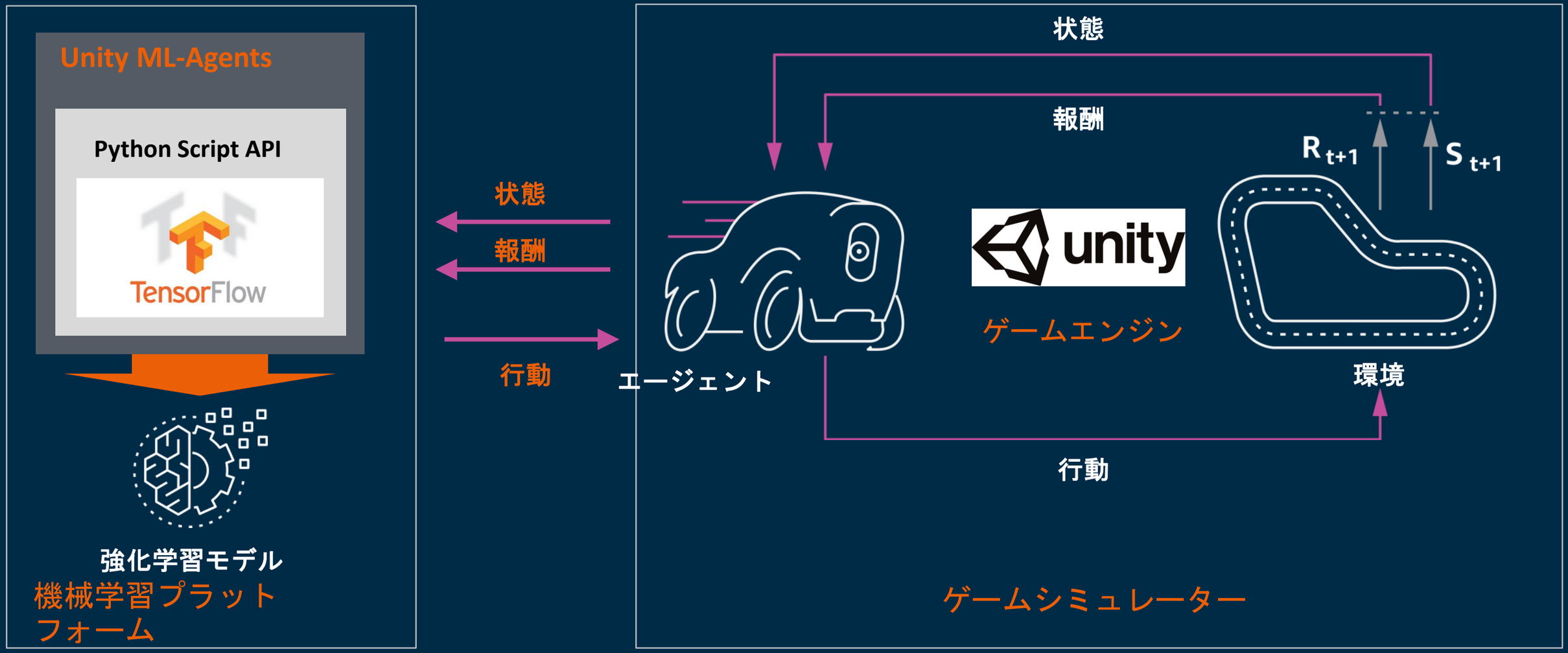
In Partnership with intel

AWS DeepRacer のシミュレーターアーキテクチャ



In Partnership with intel

ゲームのシミュレーターアーキテクチャ



In Partnership with intel

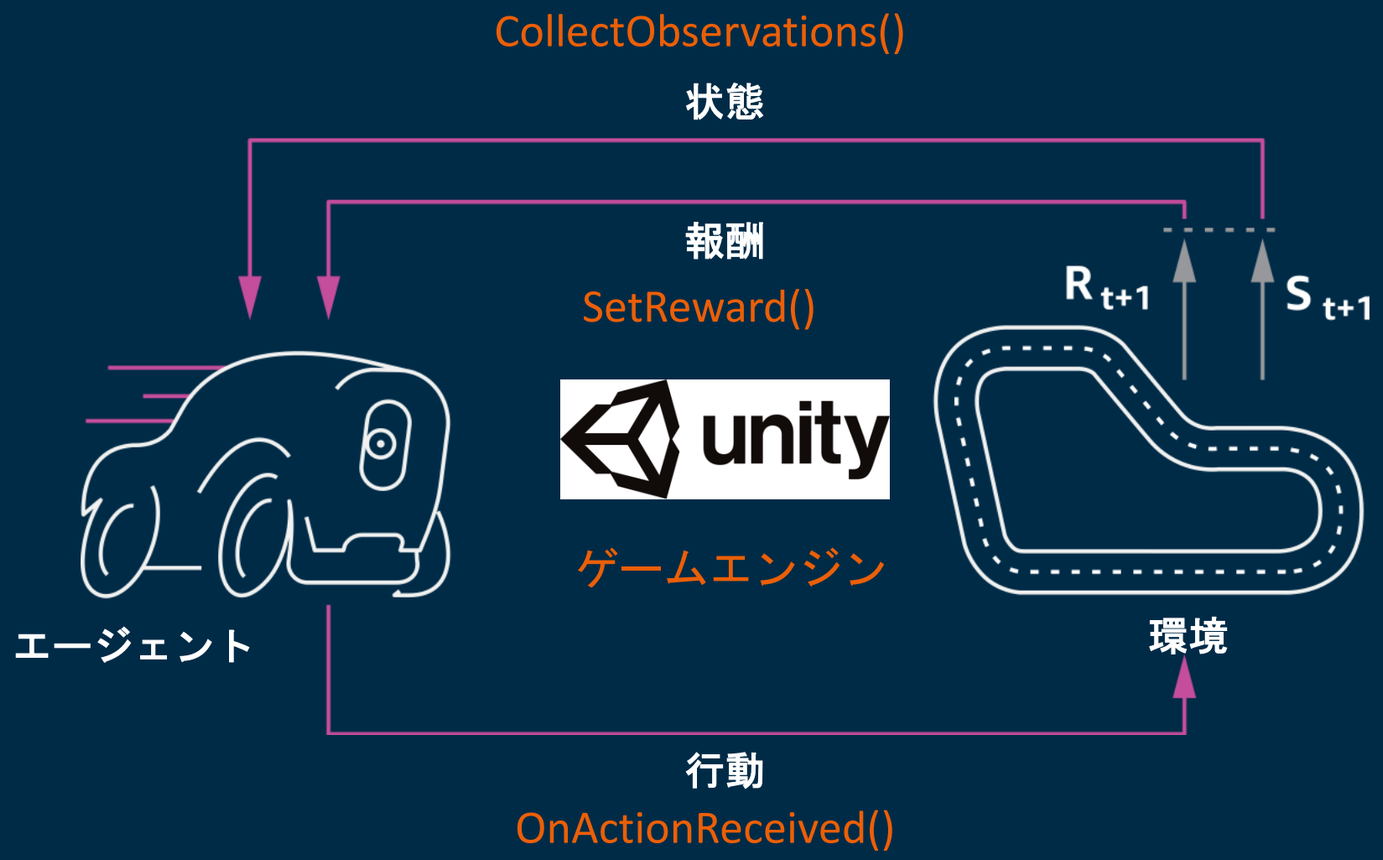
Unity をシミュレーターとして利用する理由

- Unity は Unreal Engine とともにゲーム業界で広く使われているゲームエンジン
- Unity によるとPC/コンソール/モバイルゲームに渡って50%以上のゲームが Unity 上で開発されている
- Unity には他のゲームエンジンにはない Unity ML-Agents という強力なオープンソースの機械学習ツールが用意されている

- **Unity ML-Agents (**Unity Machine Learning Agents Toolkit**)**
  - Unity のゲームやシミュレーションをAIエージェントの学習環境として提供するためのオープンソースプロジェクト。エージェントは、強化学習、模倣学習、 neuroevolution、またはそのほかの機械学習手法を簡単に利用できるPython APIを介して学習できる。

Unity ML-Agents の動作原理と API

ML-Agents Release 1以降のバージョンの API



Unity ML-Agents の代表的なAPI

- OnEpisodeBegin()
- CollectObservations(VectorSensor sensor)
- OnActionReceived(float[] vectorAction)
- SetReward()
- EndEpisode()

これらの API を既存の Unity ゲームに組み込むだけで AI エージェントを動かすことができる

In Partnership with intel

Agent が環境から取得する状態の観測方法 3種類

Vector Sensor

Camera Sensor

Ray Perception Sensor

Vector observations

- 数値型の状態情報
- データ型として「離散」と「連続」のいずれかが選択可能
- CollectObservations() メソッドを必ず定義して、これを介して状態を取得する
- ゲーム環境の設定が変われば CollectObservations() を定義し直さなければならない場合がある

Visual observations

- Unityのカメラオブジェクトを介して視覚的観測をエージェントに提供する状態取得方法
- CollectObservations() メソッドは利用しないので定義する必要はない
- カメラセンサーが自動的にピクセル情報を取得して状態を検出する
- 通常、vector observationに比べて学習に長時間かかる傾向にある

Ray cast observations

- レイキャストと呼ばれる光線ベースの物体検知方法を利用
- エージェントからレイキャスト上にある物体を検知して環境の状態を取得する
- こちらもカメラセンサーによる検知方法と似ている

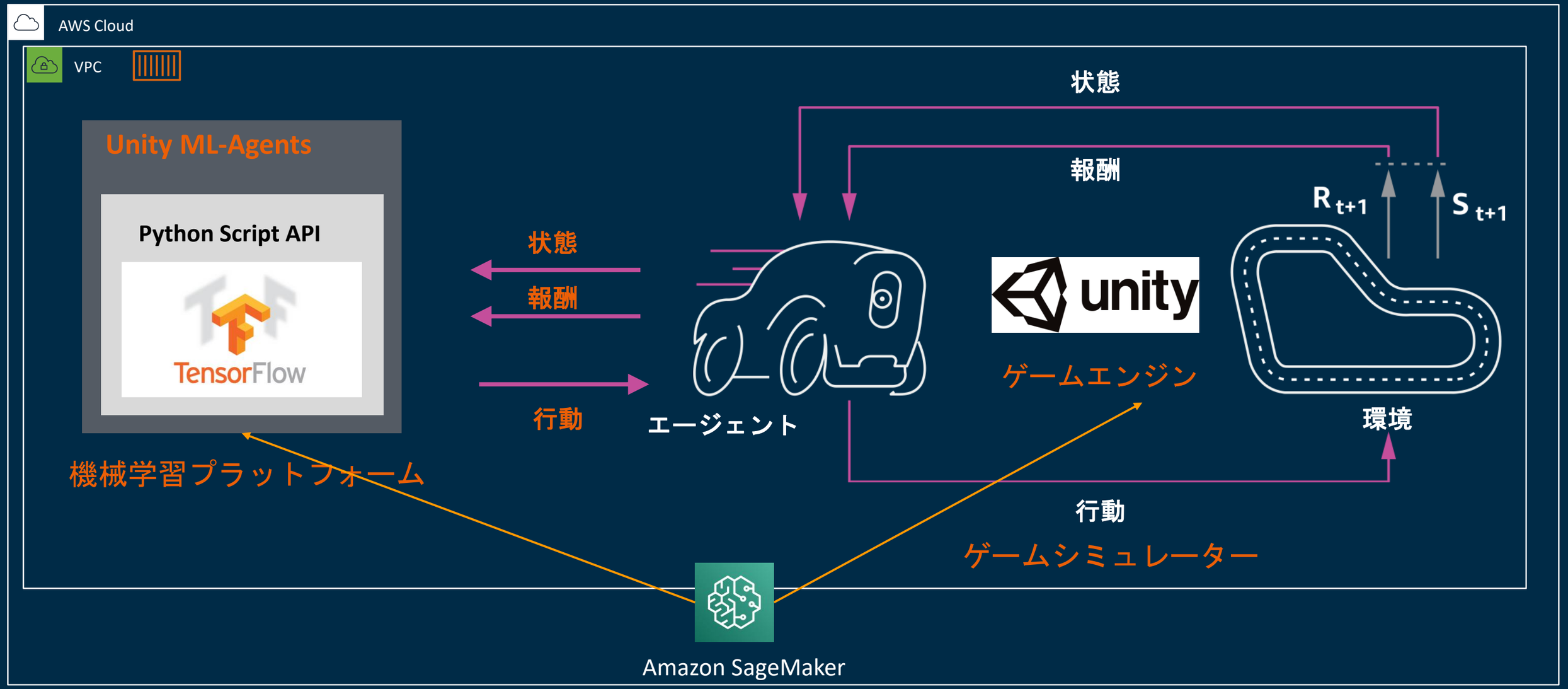
In Partnership with intel

aws DEV DAY
ONLINE JAPAN

Unity のシミュレーションを Amazon SageMaker RL で実行する

In Partnership with intel

Unity 環境を Amazon SageMaker RL 上で動かす



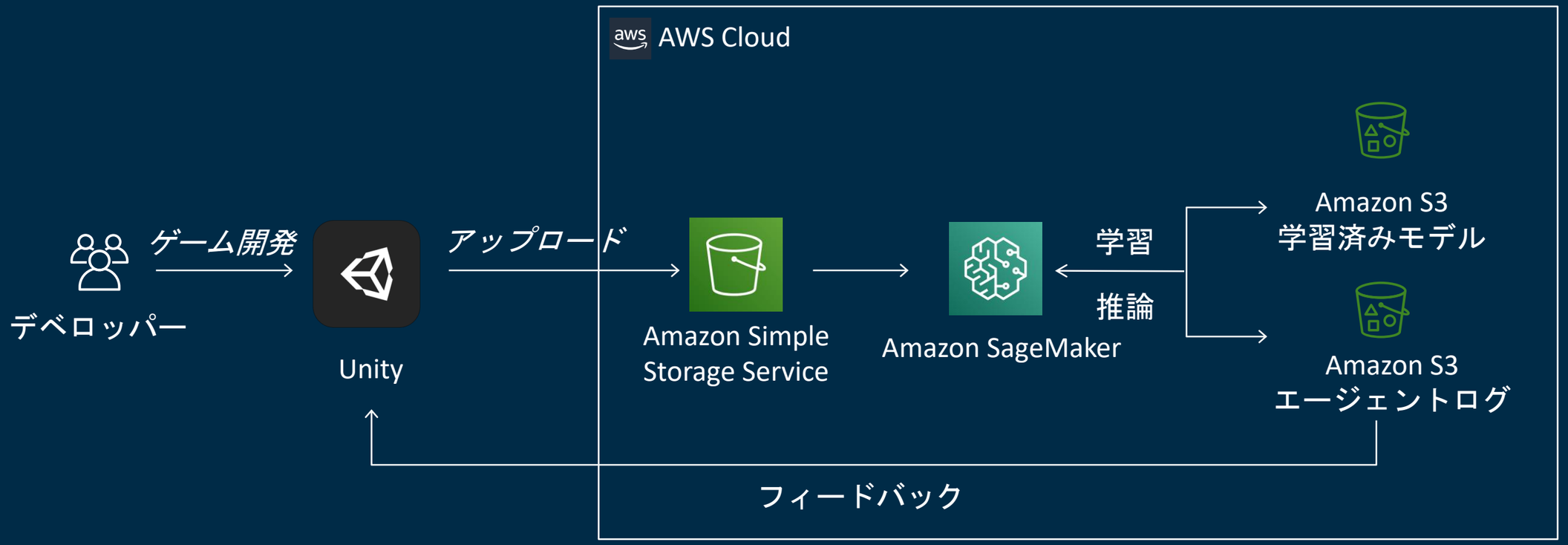
In Partnership with intel

Unity ML-Agents を SageMaker RL で動かす理由

- オンプレミスの開発マシンで動かすよりもオンデマンドで使える EC2 インスタンスでコストを削減できる
- EC2 スポットインスタンスを使えばさらにコストを下げることができる
- 強力な EC2 インスタンスタイプのラインナップを利用することにより学習やシミュレーション時間の短縮が可能
- SageMaker RLで複数のインスタンス間で分散学習を行えば学習やシミュレーション時間の短縮が可能

In Partnership with intel

Unity シミュレーターの AWS アーキテクチャ



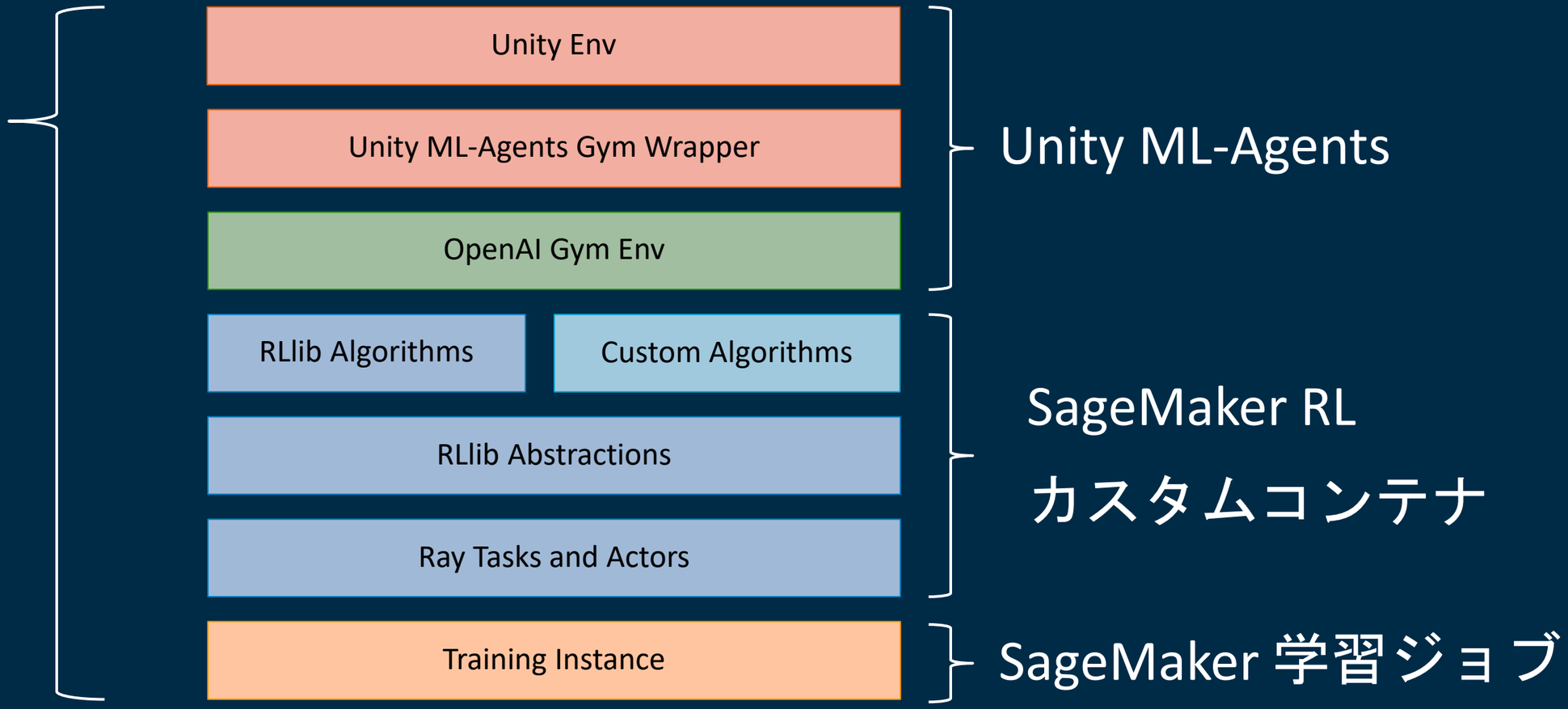
https://github.com/aws/amazon-sagemaker-examples/tree/master/reinforcement_learning/rl_unity_ray

In Partnership with intel

SageMaker RL と Unity ML-Agents との統合

SageMaker 学習ジョブ上のスタック

SageMaker

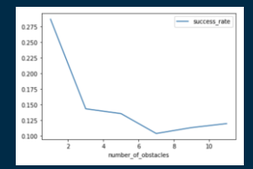


In Partnership with intel

SageMaker RLを使ったゲームテストの自動化

ほぼ全ての自動テストプロセスを SageMaker 上で完結できる

ゲーム制作 → 強化学習エージェントの学習 → モデルのデプロイ → 新規ステージ内でエージェントを走らせ評価 → ログ収集・結果を可視化



Unity SageMaker

In Partnership with intel

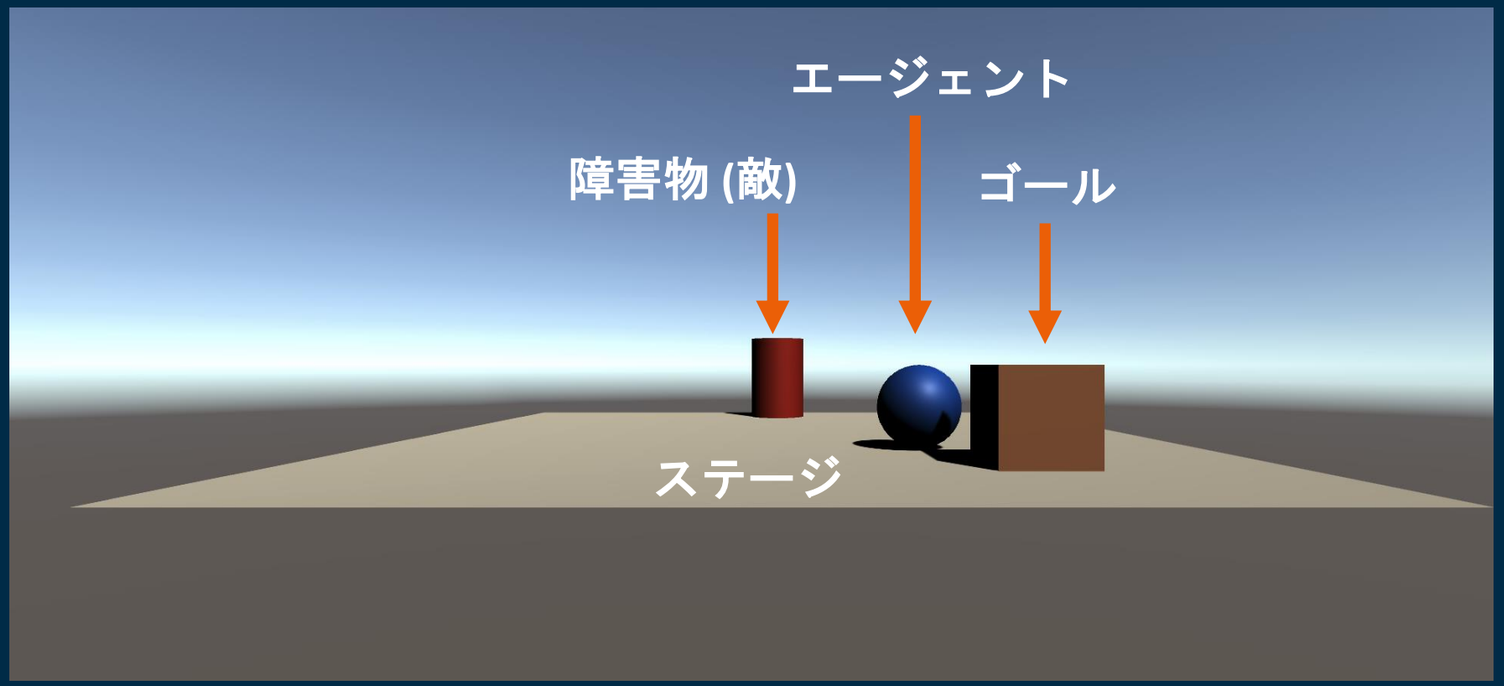
aws DEV DAY
ONLINE JAPAN

実際の QA Automation の流れについて

In Partnership with intel

PoC 用の Unity サンプルゲーム

- SageMaker RLでのテスト自動化 (QA Automation) の実証実験 (PoC) のためにUnity サンプルゲームを用意
- サンプルゲームはチュートリアルから派生したもので、比較的簡単に実装可能



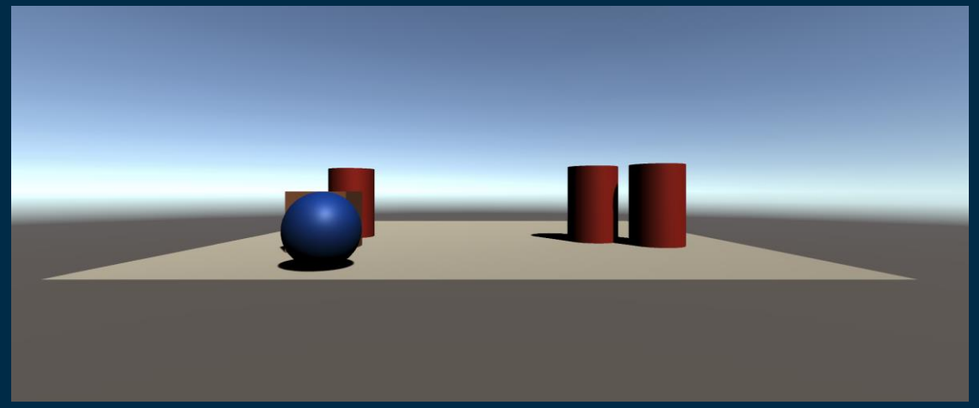
- エージェントがゴールに到達
  - +1.0 の正の報酬を獲得
  - 1ゲーム終了
- エージェントが障害物に衝突
  - -1.0 の負の報酬を取得
  - 1ゲーム終了
- エージェントがステージ上から落下
  - -1.0 の負の報酬を取得
  - 1ゲーム終了

In Partnership with intel

PoC 用の Unity サンプルゲーム

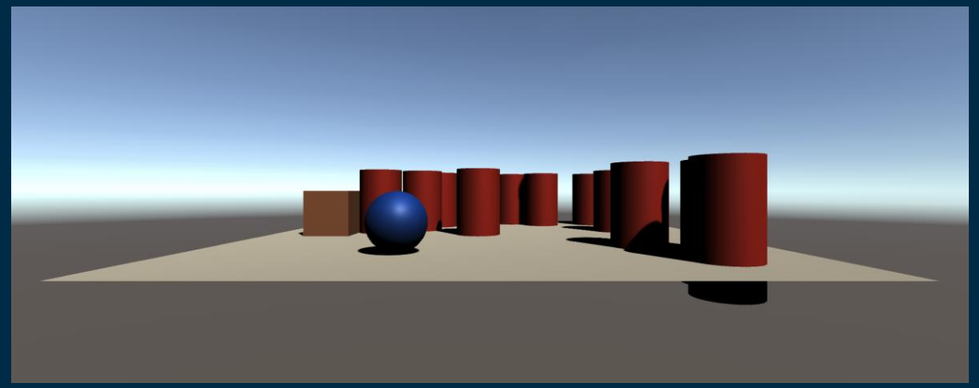
障害物の数は、ステージの難易度に応じて変化する

Obstacles = 3



低難度

Obstacles = 11



高難度

In Partnership with intel

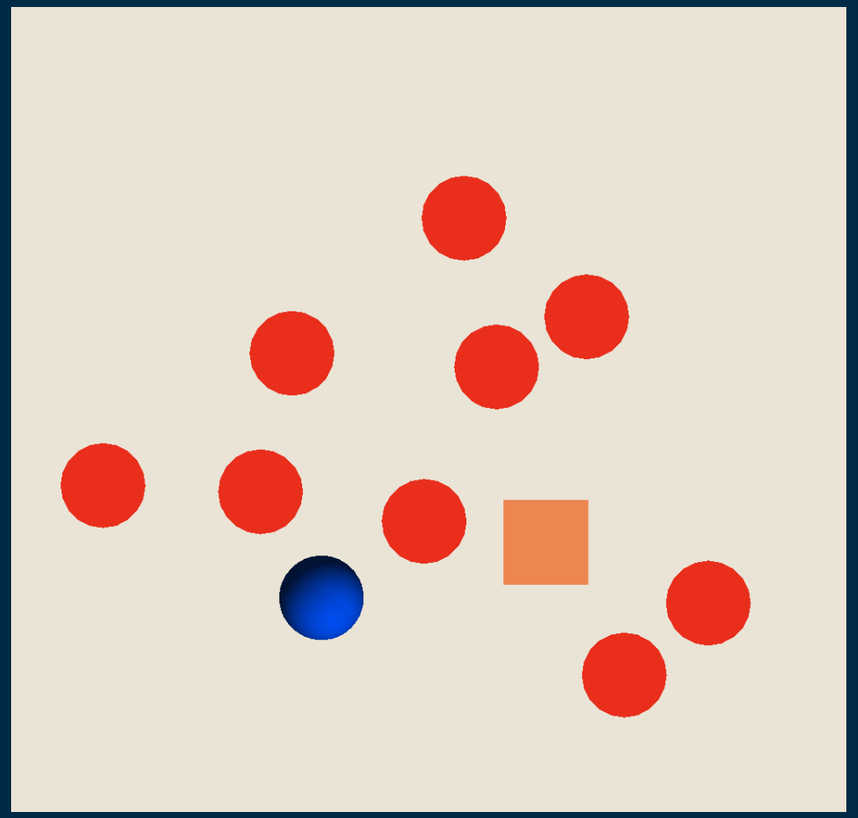
Unity のゲームステージ設定ファイル

```
{
        "training": {
                "numObstacles": [1, 2, 3, 4, 5]
        },
        "simulation": {
                "numObstacles": [1, 3, 5, 7, 9, 11]
        }
}
```

RollerBallConfig.json

- 学習時の障害物の個数とシミュレーション (ゲームテスト時) の障害物の個数を設定する設定ファイルを用意
- 複数の数値が配列に入っている場合は、エピソード毎にランダムに数値を参照してその個数分の障害物が出現
- これにより様々なパターンのステージでエージェントを学習させ、テストも様々なパターンで実施可能

エージェントは上部カメラを通して環境を観測



- エージェントは上部カメラを通して物体の形状と色彩で環境を認識する
- エージェントの状態観測にカメラセンサーによるVisual Observationの方式をとったのは、障害物の数の変化による環境変動に対してCollectObservations() メソッドを再定義することなく柔軟に対応できるから

実際のゲームテストの自動化のステップ

ほぼ全ての自動テストプロセスをSageMaker上で完結できる

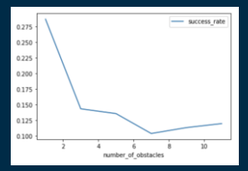
ML-Agents APIをゲームへ実装

強化学習エージェントの学習

モデルのデプロイ

新規ステージ内でエージェントを走らせ評価

ログ収集・結果を可視化

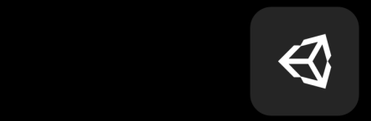


Unity

SageMaker

In Partnership with intel

ML-Agents API を Unity ゲームに組み込む



```
void Start ()
{
        rBody = GetComponent<Rigidbody>();
        if (IsTrainingMode() == false)
        {
                string datetimeStr = System.DateTime.Now.ToString("yyyyMMdd_HHmmss");
                string filePath = Application.dataPath + "/Logs";
                if (!Directory.Exists(filePath)) Directory.CreateDirectory(filePath);
                string logFile = filePath + "/" + datetimeStr + logName;
                sw = new StreamWriter(logFile, true, Encoding.UTF8);
                // header
                SaveLog(0);
        }
}
```

Start() はUnityのゲームシーンのスタート時に一度コールされる

- Unity のゲームシーンの起動時の初期化処理
- シミュレーションの際のエージェントの状態のログを書き出すための準備を実施

In Partnership with intel

ML-Agents API を Unity ゲームに組み込む

```
public override void OnEpisodeBegin()
{
        if (this.transform.localPosition.y < 0)
        {
                // If the Agent fell, zero its momentum
                this.rBody.angularVelocity = Vector3.zero;
                this.rBody.velocity = Vector3.zero;
                this.transform.localPosition = new Vector3( 0, 0.5f, 0);
        }

        // Move the target to a new spot
        Target.localPosition = new Vector3(Random.value * 8 - 4, 0.5f, Random.value * 8 - 4);
        obstacle.ResetObstacles();
        episodeCount++;
}
```

OnEpisodeBegin() は一回のゲームステージ (1ゲームプレイ) が開始されるたびにコールされる

- OnEpisodeBegin() は一回のゲームステージ (1ゲームプレイ) が開始されるたびにコールされる
- ここでは、任意の個数の障害物とゴール一つをランダムな場所に配置する初期化処理を行っている

ML-Agents API を Unity ゲームに組み込む

```
public override void OnActionReceived(float[] vectorAction)
{
        // Actions, size = 2
        Vector3 controlSignal = Vector3.zero;
        controlSignal.x = vectorAction[0];
        controlSignal.z = vectorAction[1];
        rBody.AddForce(controlSignal * speed);

        // Rewards
        float distanceToTarget = Vector3.Distance(this.transform.localPosition, Target.localPosition);

        // Reached target
        if (distanceToTarget < 1.42f)
        {
                if (IsTrainingMode() == false) SaveLog(1);
                SetReward(1.0f);
                EndEpisode();
                return;
        }
```

OnActionReceived() はエージェントが毎ステップ (毎フレーム) 行動を起こす度にコールされる

エージェントがゴールに到達した時の条件

エージェントがゴールに到達できた際には、+1.0 の正の報酬を与える

.....

In Partnership with intel

ML-Agents API を Unity ゲームに組み込む

```
....
            // Collided with obstacles
            foreach (var actor in obstacle.obstacleObjs)
            {
                        float distanceToActor = Vector3.Distance(this.transform.localPosition, actor.transform.localPosition);
                        if (distanceToActor < 1.42f)
                        {
                                    if (IsTrainingMode() == false) SaveLog(2);
                                    SetReward(-1.0f);
                                    EndEpisode();
                                    break;
                        }
            }

            // Fell off platform
            if (this.transform.localPosition.y < 0)
            {
                        if (IsTrainingMode() == false) SaveLog(3);
                        SetReward(-1.0f);
                        EndEpisode();
            }
}
```

エージェントが障害物に衝突

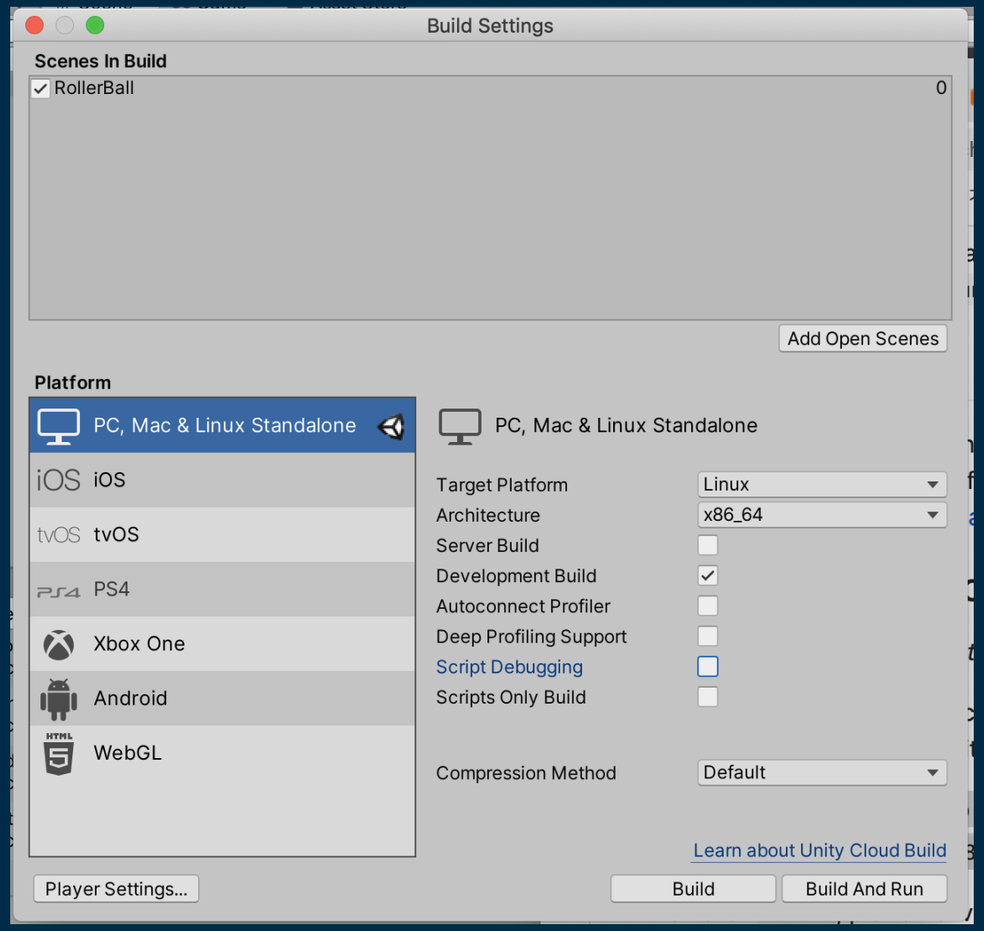
-1.0 の負の報酬を与える

エージェントがステージから落下

-1.0 の負の報酬を与える

In Partnership with intel

Unity のビルドと S3 へのアップロード



- SageMaker RL 上でゲームが実行できるように Unity をビルドする
- SageMaker コンテナが Linux で動作するためビルド時の Target Platform は Linux
- Architecture を x86_64 に設定
- ビルド後には下記のファイルが出力される
  - <environment-name>.x86_64
  - Subdirectory: <environment-name>_Data/
  - LinuxPlayer_s.debug
  - UnityPlayer_s.debug
  - UnityPlayer.so
- Unity のビルド済みのバイナリファイルを任意の S3 バケットにアップロード

In Partnership with intel

実際のゲームテストの自動化のステップ

ほぼ全ての自動テストプロセスをSageMaker上で完結できる

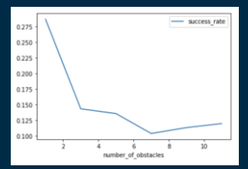
ML-Agents APIをゲームへ実装

強化学習エージェントの学習

モデルのデプロイ

新規ステージ内でエージェントを走らせ評価

ログ収集・結果を可視化



Unity

SageMaker

In Partnership with intel

SageMaker 上での RL エージェントの学習

SageMaker

```
s3_binary_path = 's3://sagemaker-unity-mlagents/RollerBall'
metric_definitions = RLEstimator.default_metric_definitions(RLToolkit.RAY)
estimator = RLEstimator( entry_point="train-unity.py",
                                                    source_dir='src',
                                                    dependencies=["common/sagemaker_rl"],
                                                    image_name=custom_image_name,
                                                    role=role,
                                                    train_instance_type=instance_type,
                                                    train_instance_count=1,
                                                    output_path=s3_output_path,
                                                    base_job_name=job_name_prefix,
                                                    metric_definitions=metric_definitions,
                                                    hyperparameters={
                                                                # Attention scientists! You can override any Ray algorithm parameter here:
                                                                "rl.training.config.env_config.env_name": "RollerBallSimulation.x86_64",
                                                                "rl.training.stop.timesteps_total": 500,
                                                    }
                                                )

estimator.fit({'train': s3_binary_path}, wait=True)
```

Unity のビルド済みのファイルを置いた S3 のパスを指定

ハイパーパラメータの設定

学習の実行開始

In Partnership with intel

実際のゲームテストの自動化のステップ

ほぼ全ての自動テストプロセスをSageMaker上で完結できる

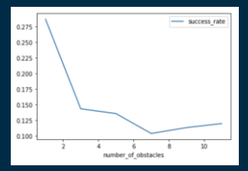
ML-Agents APIをゲームへ実装

強化学習エージェントの学習

モデルのデプロイ

新規ステージ内でエージェントを走らせ評価

ログ収集・結果を可視化



Unity

SageMaker

In Partnership with intel

ゲームステージを評価する

SageMaker

```
%%time
estimator_eval = RLEstimator( entry_point="evaluate-unity.py",
                                                    source_dir='src',
                                                    dependencies=["common/sagemaker_rl"],
                                                    image_name=custom_image_name,
                                                    role=role,
                                                    train_instance_type=instance_type,
                                                    train_instance_count=1,
                                                    output_path=s3_output_path,
                                                    base_job_name=job_name_prefix + "-evaluation",
                                                    hyperparameters={
                                                                "evaluate_episodes": 1000,
                                                                "algorithm": "PPO",
                                                                "env": "RollerBallSimulation.x86_64"
                                                    }
                                          )

estimator_eval.fit({'model': checkpoint_path, 'train': s3_binary_path}, wait=True) job_name = estimator_eval.latest_training_job.job_name
```

S3に保存された学習済みのモデルを指定する

評価・テストしたいビルド済みの
ゲームステージの S3 パスを指定する

aws

In Partnership with intel

実際のゲームテストの自動化のステップ

ほぼ全ての自動テストプロセスをSageMaker上で完結できる

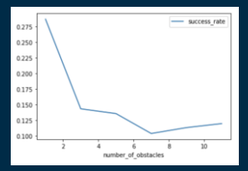
ML-Agents APIをゲームへ実装

強化学習エージェントの学習

モデルのデプロイ

新規ステージ内でエージェントを走らせ評価

ログ収集・結果を可視化



Unity

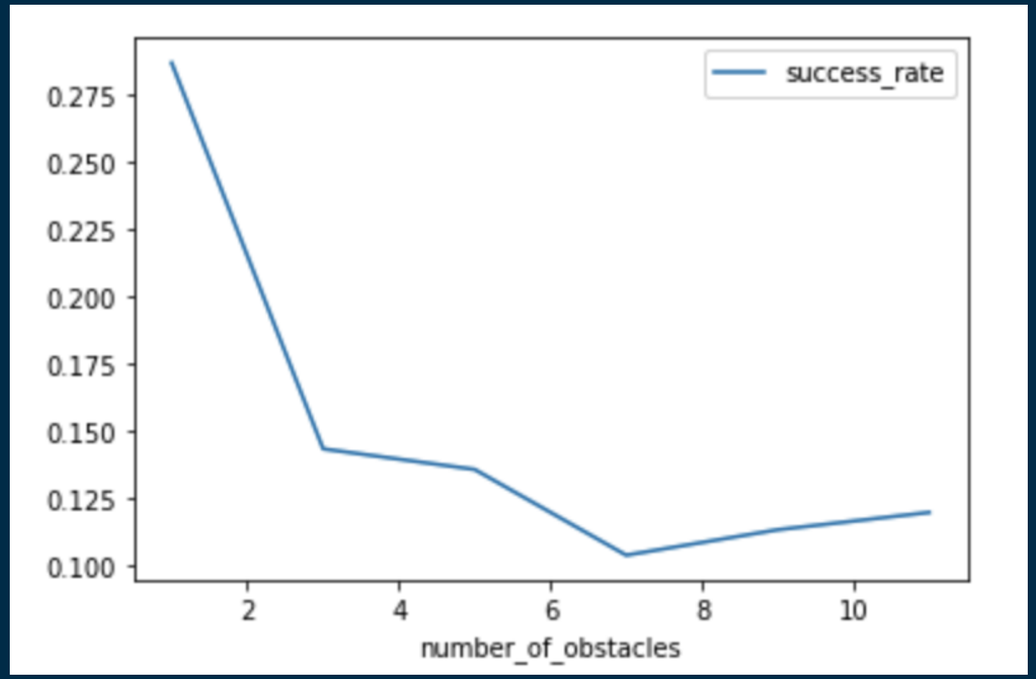
SageMaker

In Partnership with intel

ゲームステージの評価結果の可視化

SageMaker

	number_of_obstacles	success_rate	total_count
4	1	0.287009	331
3	3	0.143369	279
5	5	0.135714	280
2	7	0.103806	289
0	9	0.113269	309
1	11	0.119718	284



Success rate (N) = (N個の障害物の時のエージェントのゴール到達成功回数) / (N個の障害物の時のトータルの試行回数)

- RLエージェントによる自動テストの結果を Pandas などで集計・分析してインサイトを得る
- 成功率 (Success rate) は、障害物の数に応じて下がっていく
- 成功率は、障害物の数が1から３に変化した時に急激に落ち込む
- 障害物が7個以上の場合は、成功率にそれほど変化がない

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

Demo

In Partnership with intel

まとめ

- ゲーム開発において、強化学習を用いたAIによる自動テストがトレンドになってきている
- 主流なゲームエンジンでは、Unity とその機械学習ツールである Unity ML-Agents がゲームの自動テストに利用できる
- SageMaker RL 上で Unity ML-Agents の動作が可能となったことにより、Unity のゲームのシミュレーションをAWSクラウドで実行可能
- Unity の動作基盤となる SageMaker カスタムコンテナとそのスクリプトは、github の amazon-sagemaker-examplesにて公開中
- 今回紹介した PoC 用の Unity サンプルゲームは、近日中にaws-samplesで公開予定

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

Thank you!

Zenta Hori
zenthori@amazon.co.jp
@HoriZenta

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

Please complete the session survey in the mobile app.

In Partnership with intel