rclUE
a ROS2 client library for Unreal Engine

https://github.com/rapyuta-robotics/rclUE
https://github.com/rapyuta-robotics/RapyutaSimulationPlugins
https://github.com/rapyuta-robotics/turtlebot3-UE

会社概要

- ‘14年7月 ETH Zurich Spin-offとして誕生
- 100名, 20以上の国籍

link

主な投資家（総資金調達額40億円）

monoful YASKAWA SONY

取締役メンバー

Dr. Gajan Mohanarajah
Co-founder and CEO
ETH zürich

Arul Krishnamoorthy
Co-founder and CFO
NOMURA | Columbia University

Yuji Hirokawa
External Director
redhat Sw

Contents

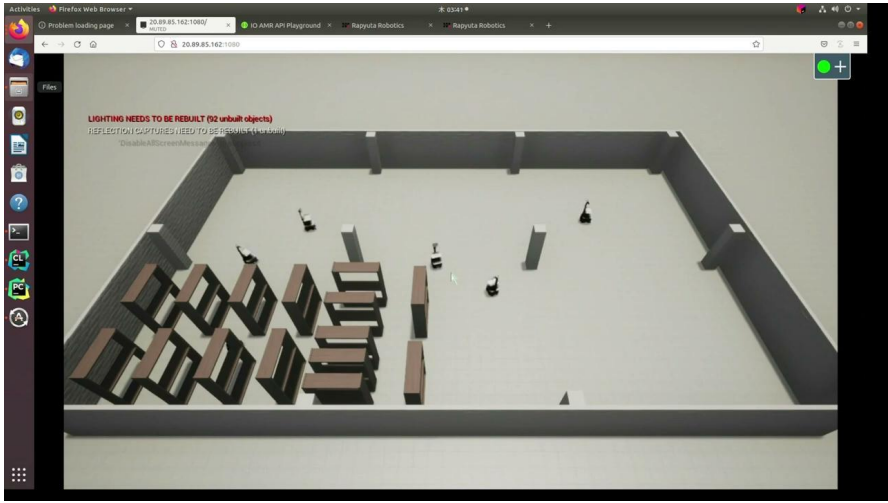
Target Simulation

ロボットシミュレータの利用目的:

- アルゴリズム開発、テスト、Offlineプログラミング, and etc.

Unreal Engine 4(UE4) & rclUE & ROS2 では？

- 複数ロボット
  - 将来的には工場全体・物流倉庫全体
- Unreal Engine4(UE4)の機能を活用
  - レンダリング
  - BluePrint Scripting Language
  - ネットワークゲーム向けの通信機能
- 精密な摩擦や接触等は対象としない?



video link

rapyuta robotics

Simulation 比較

	OSS	Physics Engine	ROS2 support	Interface/language
Gazebo	Yes	ODE, Bullet, SimBody, DART	gazebo_ros_pkg	C++
Ignition	Yes	DART	ros_ign_bridge	C++
Webots	Yes	ODE(fork)	webot_ros2	C, C++, Java, Matlab, Python
Nvidia Isaac Sim	No	PhysX	Isaac SDK	C++, Python
Unity	No	PhysX, DOTS, Havok	ROS-TCP-Connector ROS-TCP-Endpoint	C#
Unreal Engine	Yes	PhysX, Chaos	**rclUE**	C++(UE style)

ref: A Review of Physics Simulators for Robotic Applications
Robotics Simulator -wiki-

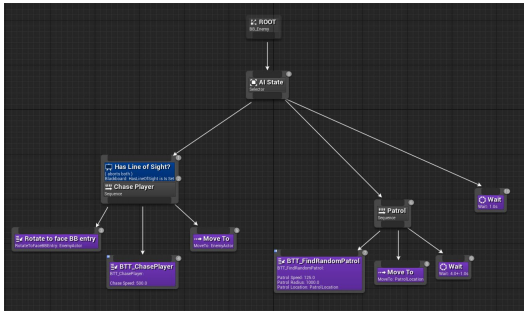
rapyuta robotics

Unreal Engine 4

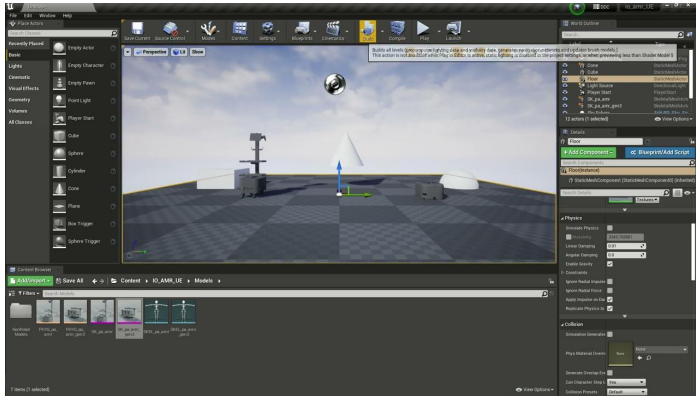
- フォトリアリスティックなレンダリング機能
  - デモ・マーケティング
  - 機械学習
- 豊富なゲーム用の model
- 物理エンジン: PhysX*
- 高機能なエディタ:
  - GUIで環境構築 → シミュレーション環境の編集
  - Blueprint → 簡易なロジックの作成
  - Behaviour Tree作成機能 → 人のシミュレーション

FF IIV Remake
https://www.jp.square-enix.com/ffvii_remake/about/index.html

*UE5 uses Chaos physics engine
*[UE採用ゲームリスト]



Behaviour tree editor
https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/ArtificialIntelligence/BehaviorTrees/BehaviorTreesOverview/

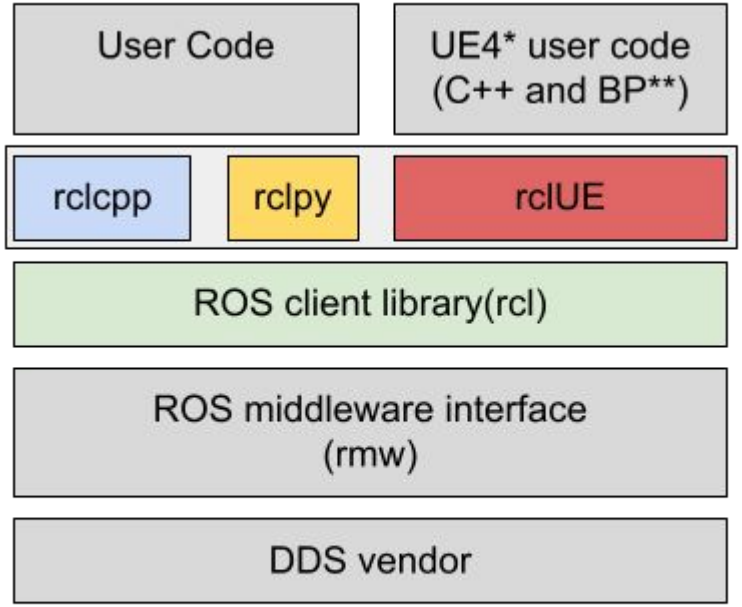


Editor and Blueprint

rapyuta robotics

rclUE is a ROS2 client library for UE4

- Supports: Ubuntu 20.04, ROS2 Foxy, UE4.27
- https://github.com/rapyuta-robotics/rclUE
- bridge node/rosbridgeを使わない
- rclcpp ではなくrclcから実装
  - Unreal Engine C++ style
- Blueprintからアクセスが可能・UE4から直接利用可能
- Actor(=UE内のオブジェクト)としてROS Componentを作成可能



*UE4: Unreal Engine 4
**BP: Blueprint visual scripting

rapyuta robotics

[C++]Publisher

Topic名、周期、MsgType設定

周期アップデート関数

```cpp
// Called when the game starts or when spawned
void AROS2PublisherNode::BeginPlay()
{
    Super::BeginPlay();
    Init();

    // Create String publisher
    StringPublisher = NewObject<UROS2Publisher>(this, UROS2Publisher::StaticClass());
    StringPublisher->RegisterComponent();
    StringPublisher->TopicName = FString("test_topic");
    StringPublisher->PublicationFrequencyHz = 1;
    StringPublisher->MsgClass = UROS2StringMsg::StaticClass();
    StringPublisher->UpdateDelegate.BindDynamic(this, &AROS2PublisherNode::MessageUpdate);
    AddPublisher(StringPublisher);
    StringPublisher->Init(UROS2QoS::DynamicBroadcaster);

    // Msg data. You can change from editor.
    // Fill default value if value is empty
    Message =  Message.IsEmpty() ? FString("Hello from C++") : Message;

}

// MessageUpdate function is called with PublicationFrequencyHz
void AROS2PublisherNode::MessageUpdate(UROS2GenericMsg *TopicMessage)
{
    UROS2StringMsg *StringMessage = Cast<UROS2StringMsg>(TopicMessage);
    StringMessage->Update(Message);
}
```

rapyuta robotics

[C++]Subscriber

Callback 追加

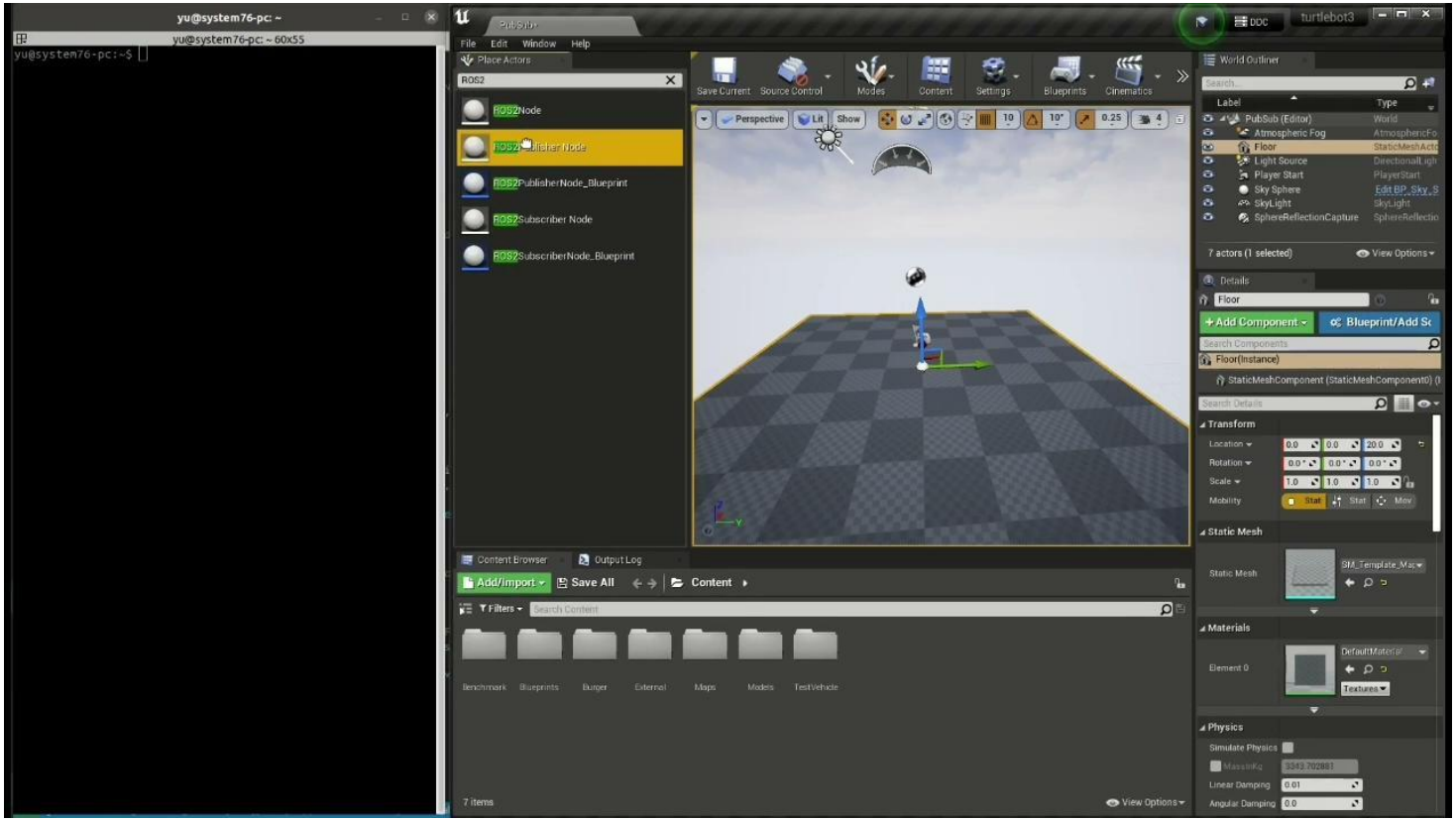
```cpp
// Called when the game starts or when spawned
void AROS2SubscriberNode::BeginPlay()
{
    Super::BeginPlay();
    Init();

    // bind callback function with topic subscription
    FSubscriptionCallback cb;
    cb.BindDynamic(this, &AROS2SubscriberNode::MsgCallback);
    AddSubscription(TEXT("test_topic"), UROS2StringMsg::StaticClass(), cb);

}

// Callback function
void AROS2SubscriberNode::MsgCallback(const UROS2GenericMsg *Msg)
{
    const UROS2StringMsg *StringMsg = Cast<UROS2StringMsg>(Msg);
    UE_LOG(LogTemp, Log, TEXT("%s"), *FString(StringMsg->GetDataData()));
}
```

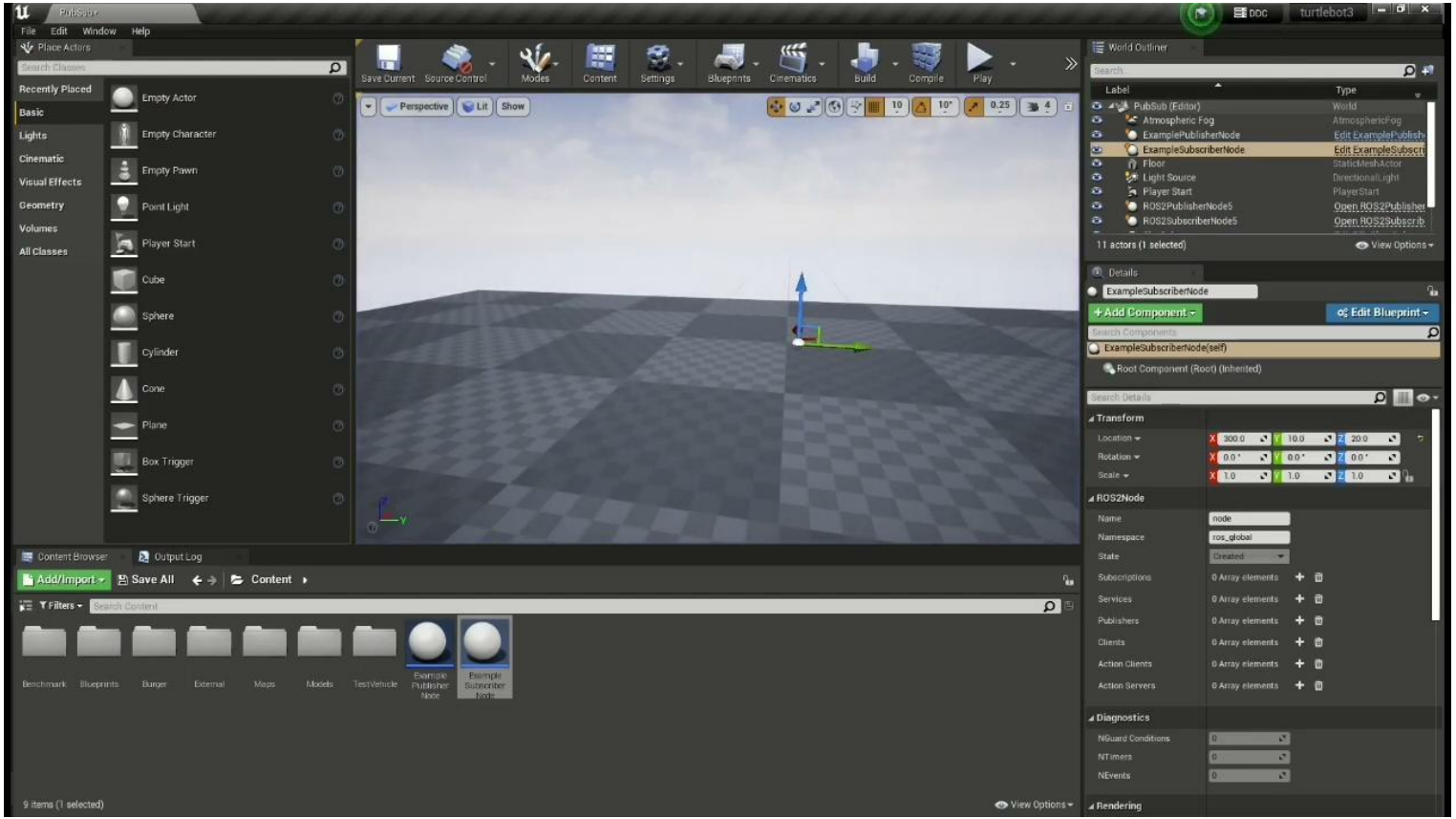
[C++]Publisher/Subscriber



video link

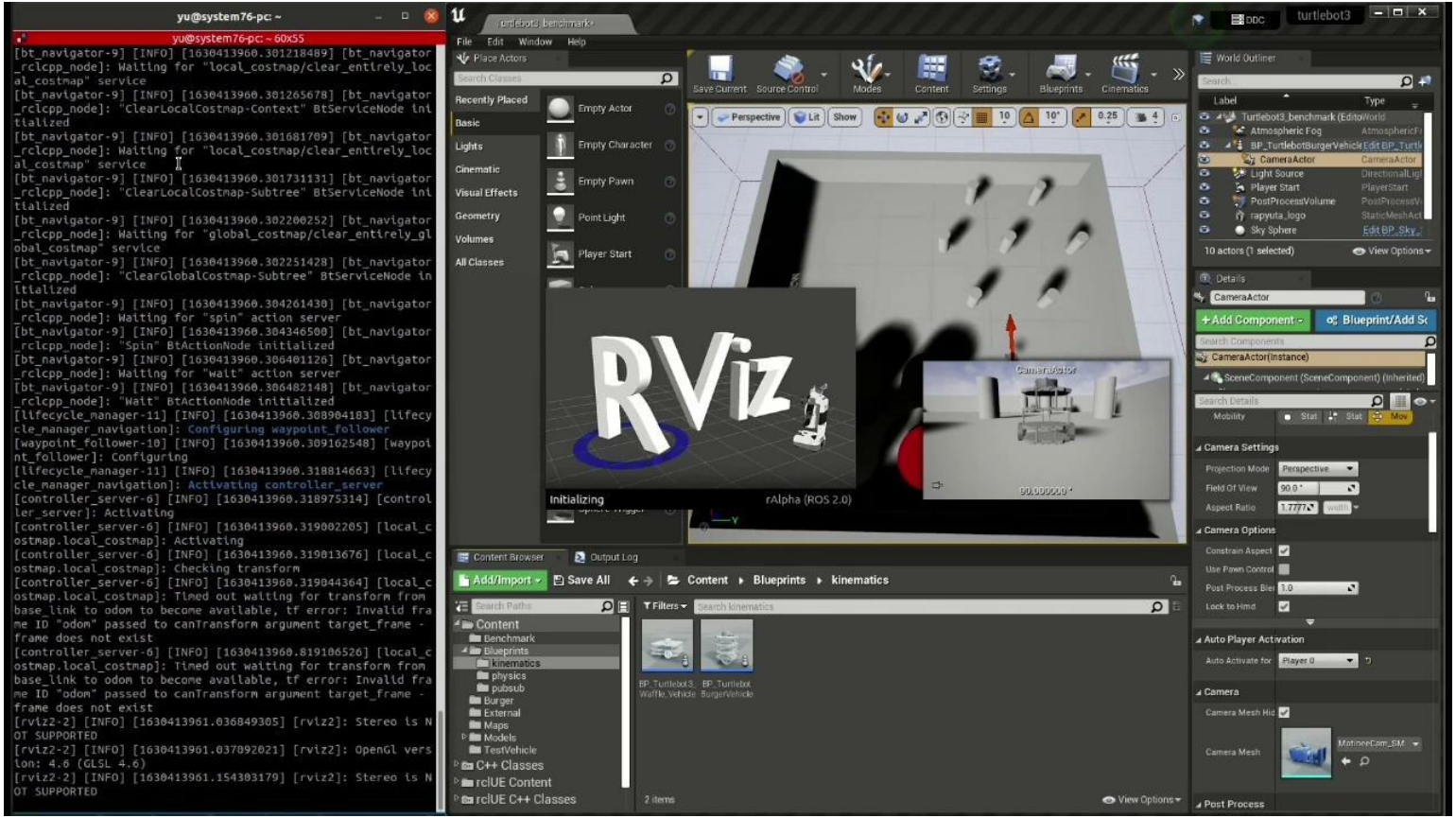
rapyuta robotics

[Blueprint]Publisher/Subscriber



[video link]

Turtlebot3



[video link]

rapyuta robotics

Summary

- rclUE によってUE4とROS2を連携させることができる。
- ROS2NodeをUE4のC++ や Bluepirntからつくれる
- UE4 のEditorを使用することで、non engineer でも環境の編集やロジックの作成が可能
  - 環境編集：倉庫や工場の作成・編集
  - シミュレーションロジックBlueprintでの作成・編集
  - Behaviour Treeにより人のシミュレーション
- ゲームで使用されているモデル等が利用可能

https://github.com/rapyuta-robotics/rclUE
https://github.com/rapyuta-robotics/RapyutaSimulationPlugins
https://github.com/rapyuta-robotics/turtlebot3-UE

Todo/注意点

Todo

- Version
  - ROS2 Galactic
  - Unreal Engine 5
- ROS
  - Robot arm
  - ROS 2 Control
  - URDF importer

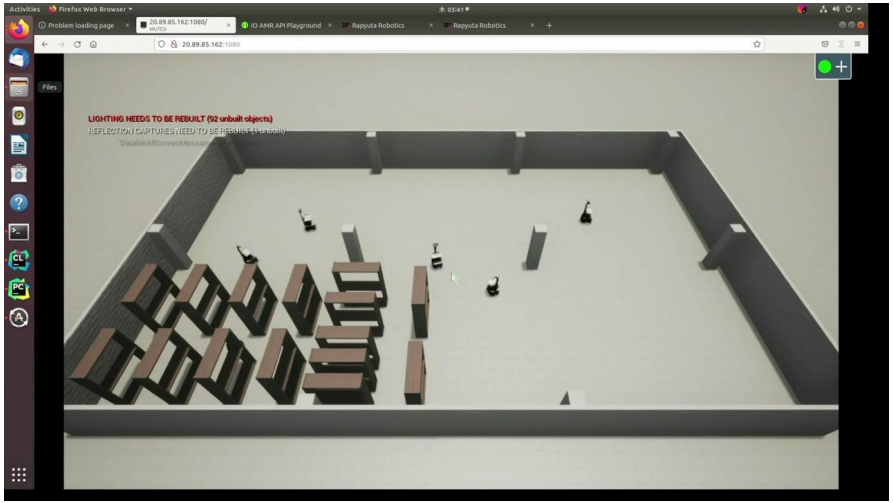
注意点

- Linux セットアップは
  - Build
    - Repoが100GB
  - Run
    - Nvidia GPU
    - 16GB-32GB Memory
- ゲーム向けなので？
  - 左手座標系
  - 単位がcm
  - Physics simulationはロボット向けのシミュレータより精確でない?

rapyuta robotics

Cloud Simulation

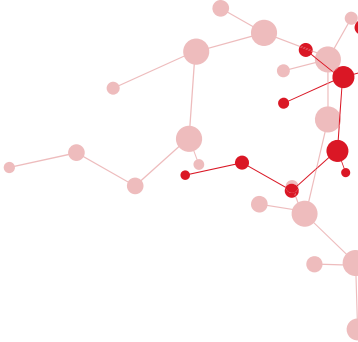
- UE4:オンライン対戦ゲームで100名のプレイヤーの実績
  - 通信システムをロボットシミュレーションに利用
- PixelStreamingでブラウザからアクセス



video link

オンラインゲーム	ロボットシミュレーション
ゲームフィールド	シミュレーション環境 倉庫・工場など
プレイヤーキャラクター	ロボット
プレイヤー	ロボット制御ソフト (navigationなど)
ゲーム視聴者	オペレータ等

rapyuta robotics



Thank you

We’re hiring and demo at booth