# Unreal Engineを用いた、駐車スペース検知のための学習データ生成
～次世代自動駐車システムの実現に向けて～

アイシン精機(株)
小久保 嘉人・丸山 大尭・浅田 修作・末次 恵久

シリコンスタジオ(株)
武藤 洋介・梅村 篤志・河野 駿介

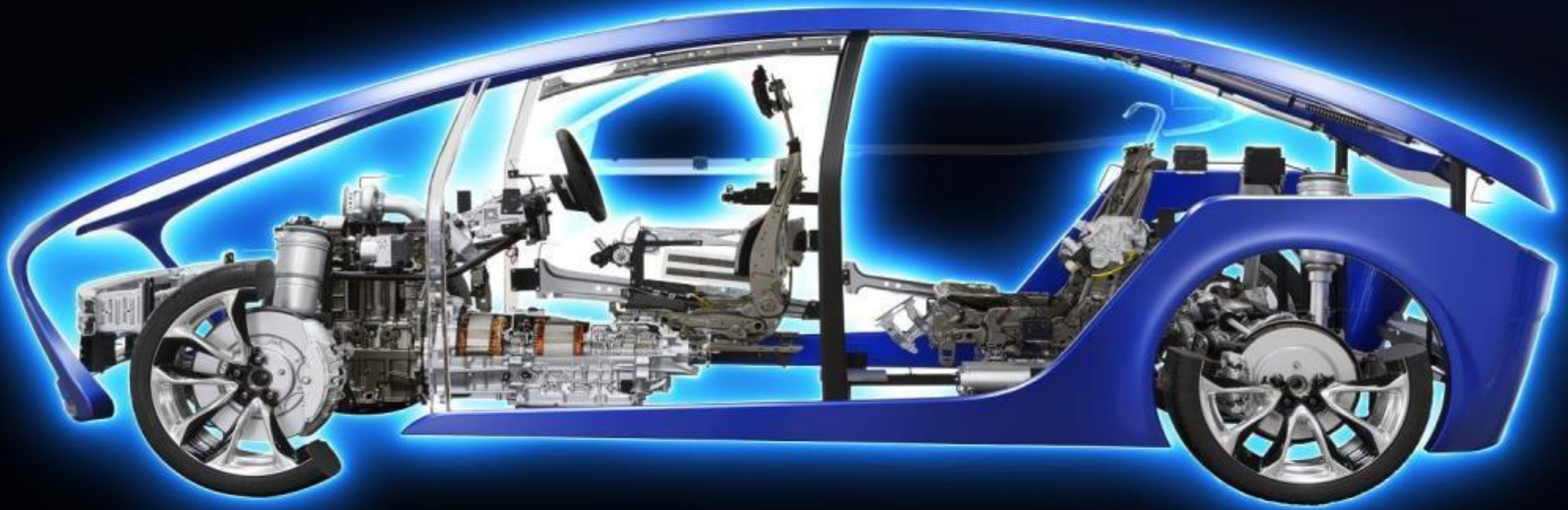
# 世界第6位の自動車部品メーカー

| | |
|---|---|
| 本社 | 愛知県刈谷市 |
| 売上高 | 4兆431億円（連結） |
| 従業員数 | 119,732名（連結） |
| 連結対象会社 | 216社<br>※日本、北米、欧州、アジア、南米 |
| 事業内容 | 自動車部品<br>/住生活・エネルギー分野 |

# アイシン精機の担う事業領域

パワートレイン

走行安全



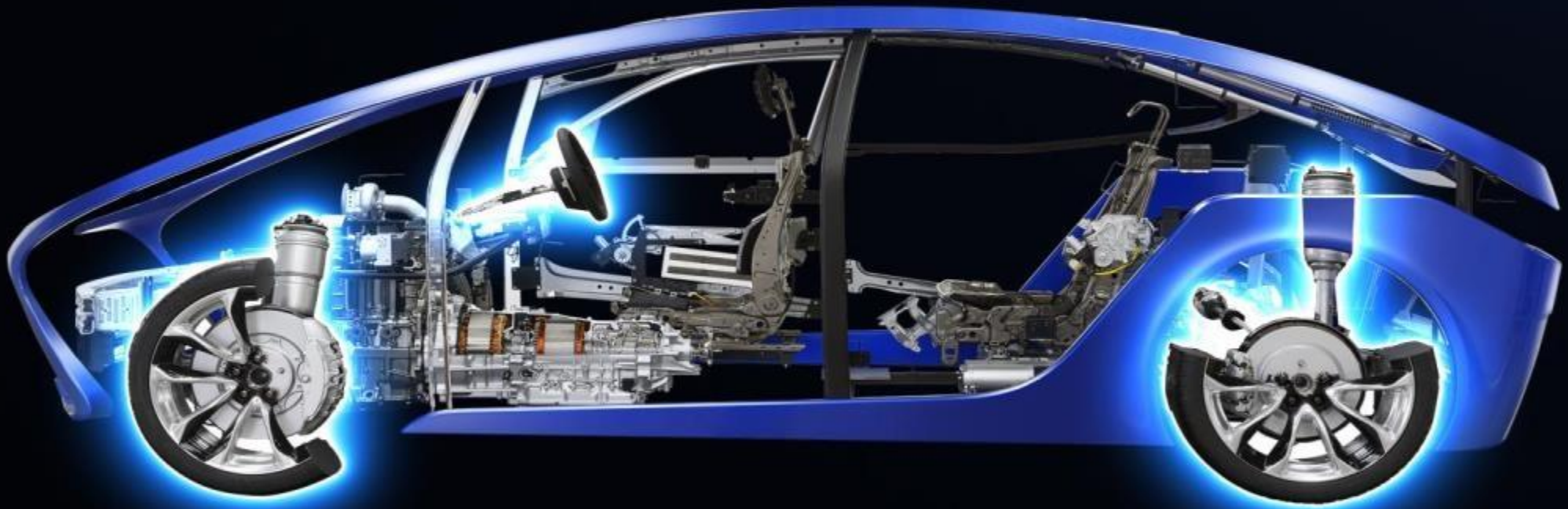
車体

情報・電子

# 自動車の『曲がる・止まる』動作の製品を担う分野

パワートレイン

走行安全



車体

情報・電子

# **走行安全領域**｜次世代自動運転システムの開発

# 目指す姿｜バレー駐車機能の実現


**目的地前で乗降を行い，駐車場へは車両が無人で移動**

⇒ 駐車場内における事故撲滅・ドライバーの移動時間削減
駐車場内スペースの利用効率化など

# 高精度な駐車枠検知の実現

# 深層学習活用課題｜アノテーション

深層学習活用のためには膨大な学習データが不可欠であり，
学習データ全てに対してアノテーション (正解値付け) が必要

1. アノテーション作業に莫大な工数が必要
2. 人手によるアノテート誤差による認識精度の低下

⇒ **学習データ生成のための駐車場シミュレータを開発**

# 駐車場シミュレータの開発｜位置付け

**自動運転機能開発においてCG画像が利用され始めている**

⇒ いずれも汎用性に富んでおり，様々な画像認識課題に対して有効

**開発シミュレータは駐車スペース特化**

⇒ 既存シミュレータでは実装されていない，駐車区画線種類やかすれ状態などを調整可能

**汎用シミュレータと併用して利用**


Microsoft Airsim

Cognata

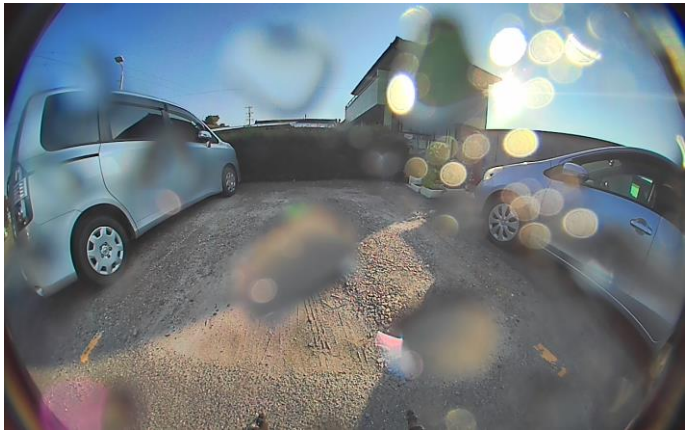

NVIDIA DRIVE Constellation

# 開発の観点｜①自動車の利用環境

**自動車の利用環境は多種多様であり，利用シーンの幅広い考慮が必要**

**天候変化や時間変化，日照条件変化が可能であること**



# 開発の観点｜②豊富な駐車枠形状

**特殊形状の駐車枠は膨大にあり，随時新たな形状が確認される可能性**

**簡単に枠種を追加/切り替え可能であること**



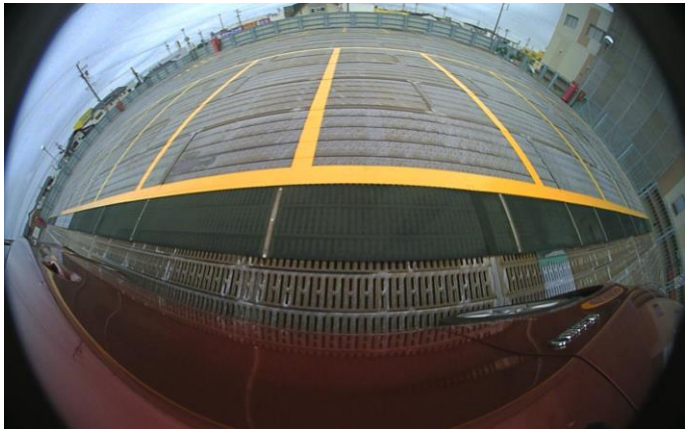



# 開発の観点｜③多様な線色/かすれ状態

**駐車枠線には多様な色/かすれ状態が存在**

**多様な枠線の色/かすれ状態を表現でき，簡易的に制御できること**






# 開発の観点｜④真値の利用判断

## 車両駐車状況によって，空き駐車スペースであると誤認する場合が存在

# 駐車場シミュレータ制作事例

AISIN GROUP

Silicon Studio

# 駐車場シミュレータ制作事例

**【アート編】**

➡ UEエディタをコンフィグレーターライクに利用

**【アノテーションデータ編】**

➡ ゲームロジックと標準機能を利用したデータ取得

※ スライド内ではアンリアルエンジンをUE,
ブループリントをBPと表記

# ワークフロー

## ① レイアウト作成

- 区画線配置
- 質感ダメージ表現

## ② 時間帯天候効果設定

- 環境光
- 路面濡れ　凹凸調整

## ③ 自車両走行パスアニメーション作成

## ④ アノテーション出力

- 駐車区画の四隅座標
- 車両IDマスク画像

# UEエディタをコンフィグレーターライクに利用

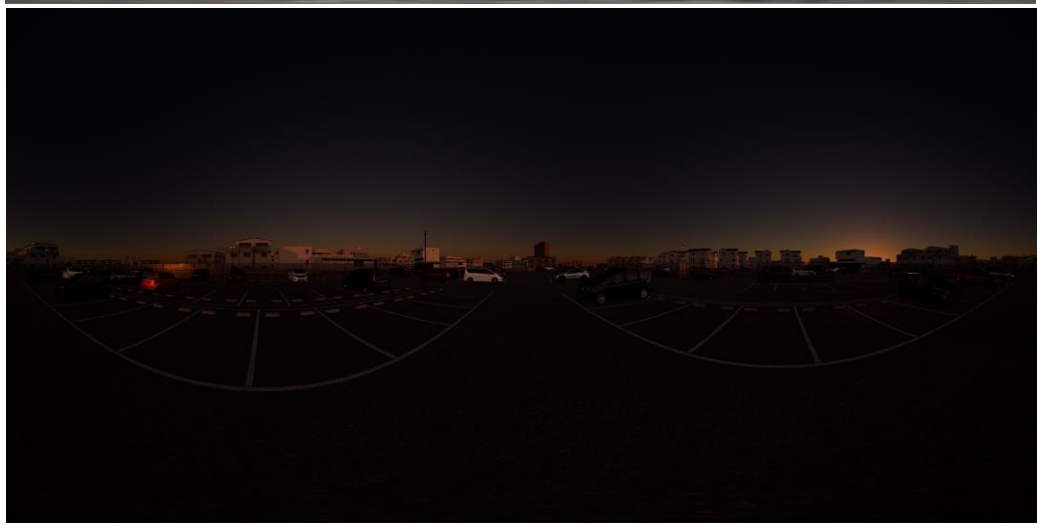
**本開発はUEを使用したコンフィグレーター**
**アプリケーションの開発とは異なる**

利用者によるシーン拡張を行えるよう
**レイアウト編集，アノテーションデータ**
**書き出しまでUEエディタ上で完結**

**必要なパラメータを露出させ，編集を用意にすることを目指した**

# UEエディタをコンフィグレーターライクに利用

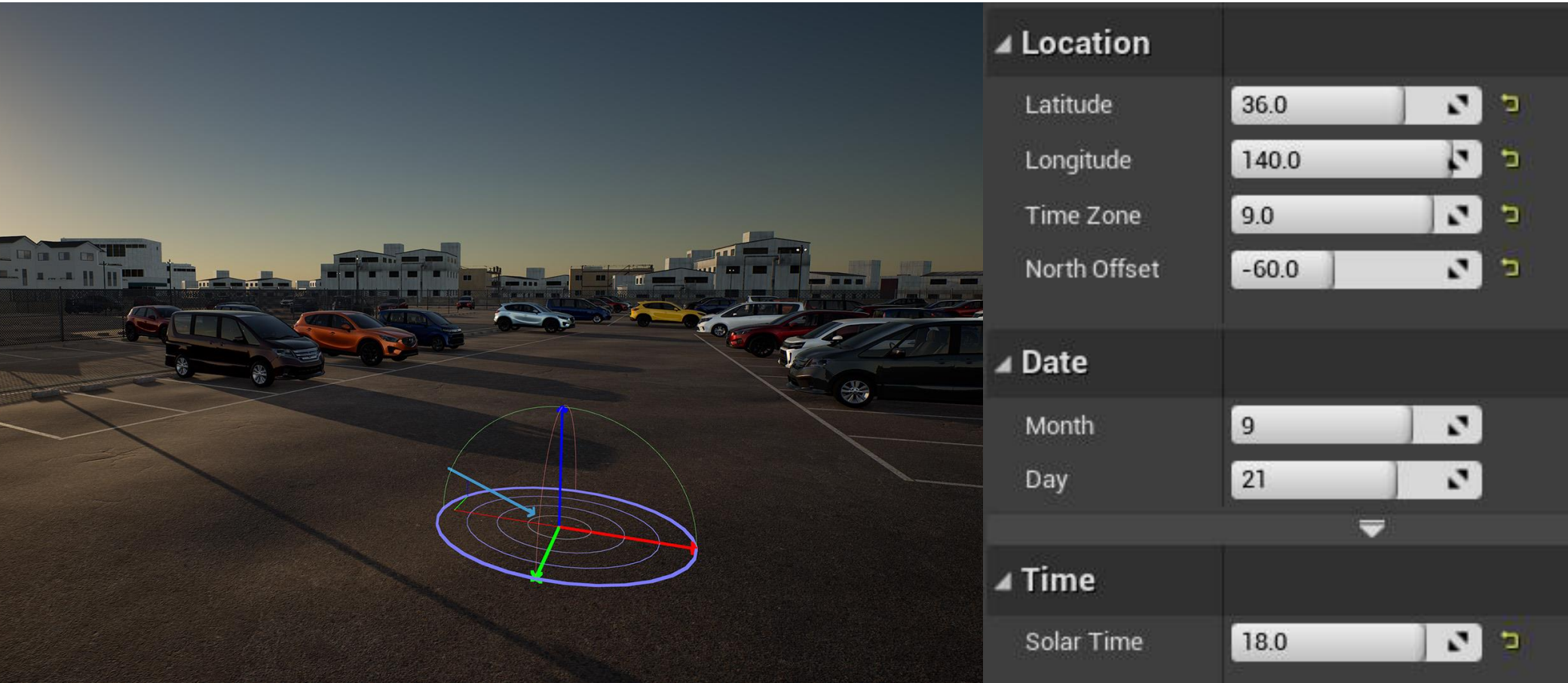
**シームレスな日照変化** ⇒ [Sky Atmosphere]を使用



# UEエディタをコンフィグレーターライクに利用

**シームレスな日照変化** ⇒ [Sky Atmosphere]を使用

・数理モデルに基づいたパラメーター
・緯度/経度/タイムゾーン/月/日/時刻を設定可能



# UEエディタをコンフィグレーターライクに利用

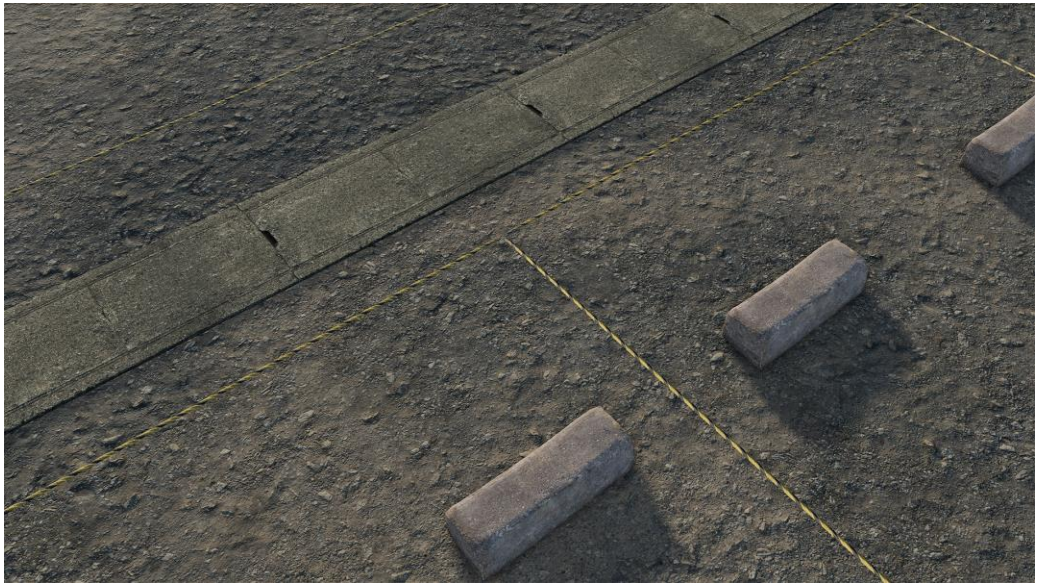
**路面凹凸変化，濡れ表現** ⇒ BPからシーン全体の質感を変更

**[Undulation]　ノーマル強度調整**

# UEエディタをコンフィグレーターライクに利用

**路面凹凸変化，濡れ表現** ⇒ BPからシーン全体の質感を変更

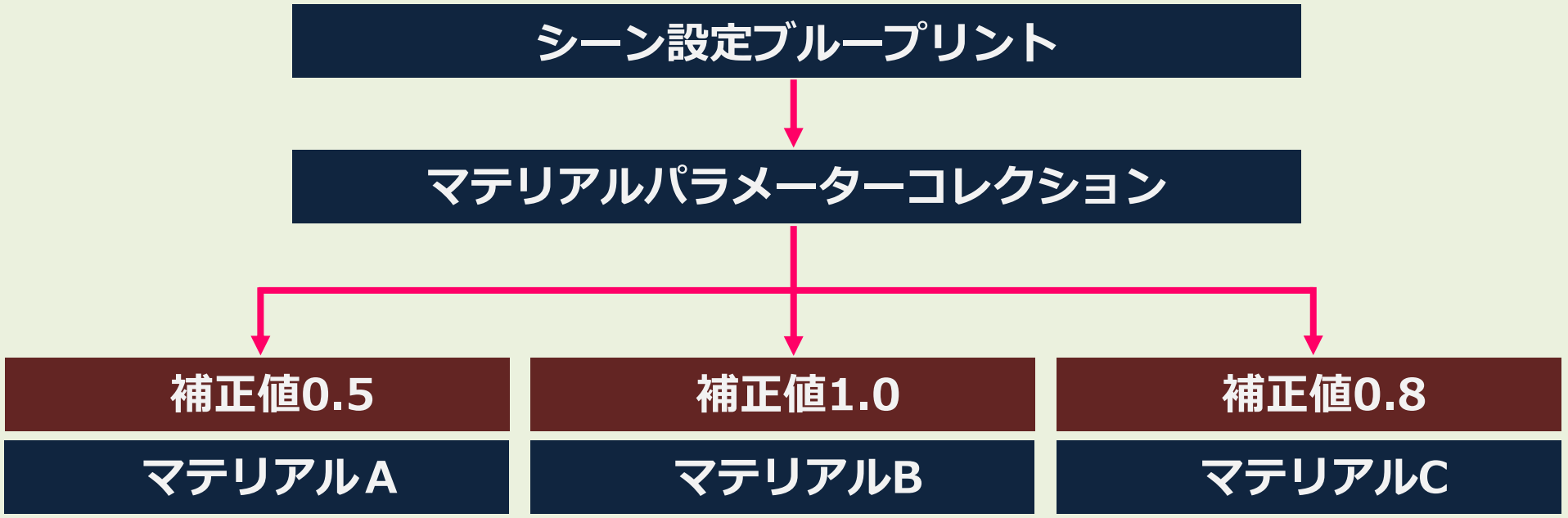
**[Wetness]　ベースカラー＆ラフネス調整**



# UEエディタをコンフィグレーターライクに利用

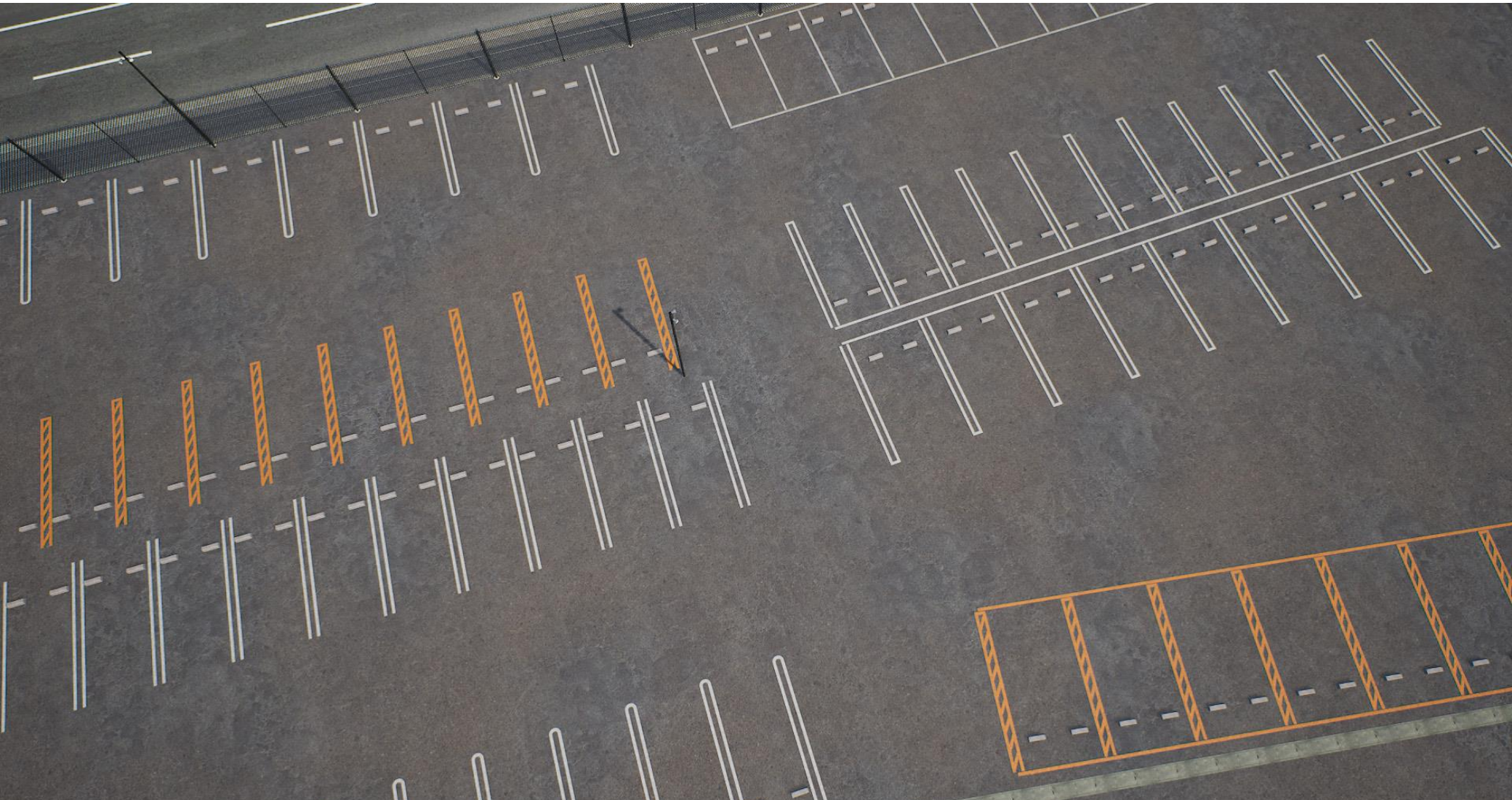
## 路面凹凸変化，濡れ表現

- BPからマテリアルパラメーターコレクションを介すことで，
  複数マテリアルを1つのUIで操作可能
- マテリアル毎に補正値を設定し，
  素材特性の違いを表現

# UEエディタをコンフィグレーターライクに利用

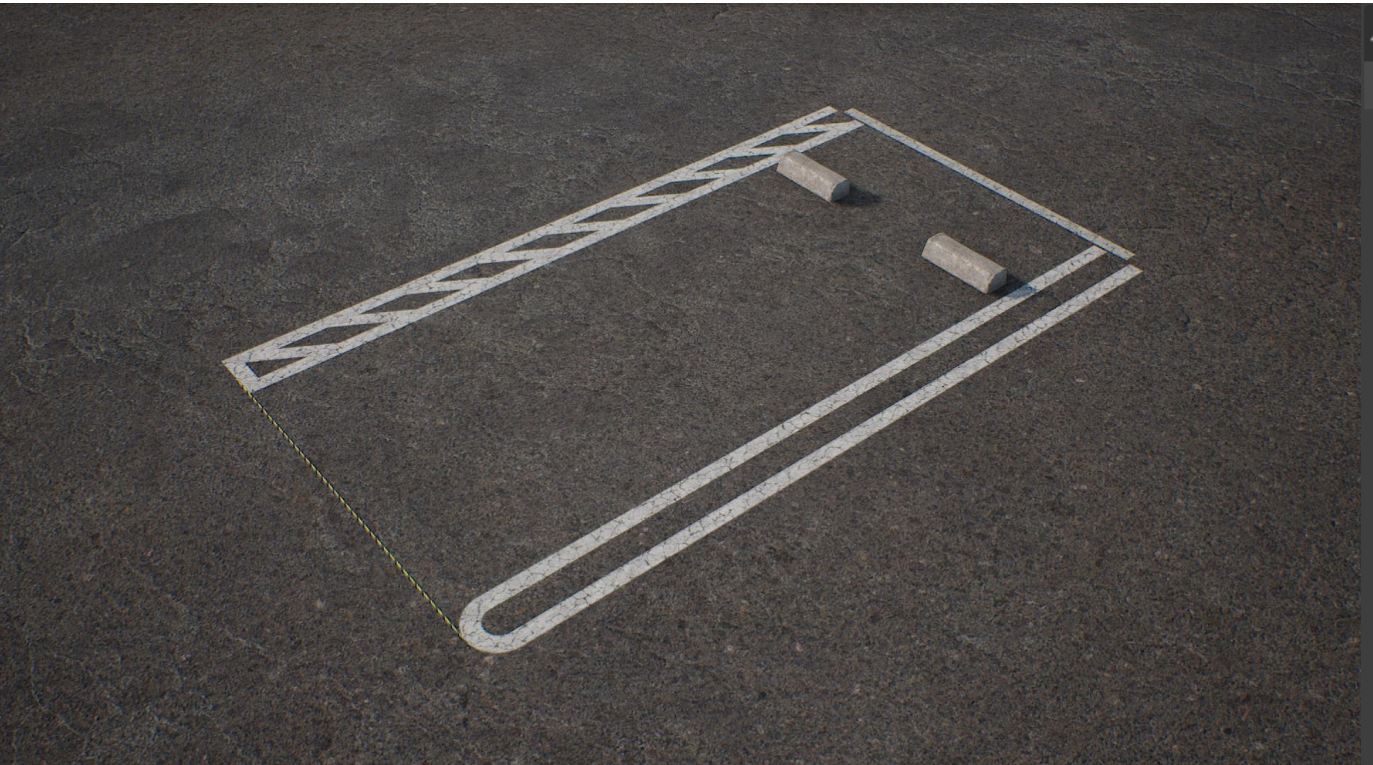
**区画線モデルのレイアウト** ⇒ BPコンストラクションスクリプトによる配置



# UEエディタをコンフィグレーターライクに利用

**区画線モデルのレイアウト** ⇒ BPコンストラクションスクリプトによる配置

① 白線/トラロープ モデル切り替え　② 配置個数，区画幅
③ 白線上下左右　車止め　ON・OFF　④ 車止めは区画幅センターに自動配置



# UEエディタをコンフィグレーターライクに利用

**白線かすれ表現** ⇒ 実物の白線を観察，劣化要因を3種に分類

**ひび割れ**　**剥離**　**摩耗**

# UEエディタをコンフィグレーターライクに利用

**白線かすれ表現** ⇒ 実物の白線を観察，劣化要因を3種に分類

**ひび割れ**　**塗膜が劣化し表面にクラックが発生**

# UEエディタをコンフィグレーターライクに利用

**白線かすれ表現** ⇒ 実物の白線を観察，劣化要因を3種に分類

**剥離**　**塗膜が剥がれ欠損**

# UEエディタをコンフィグレーターライクに利用

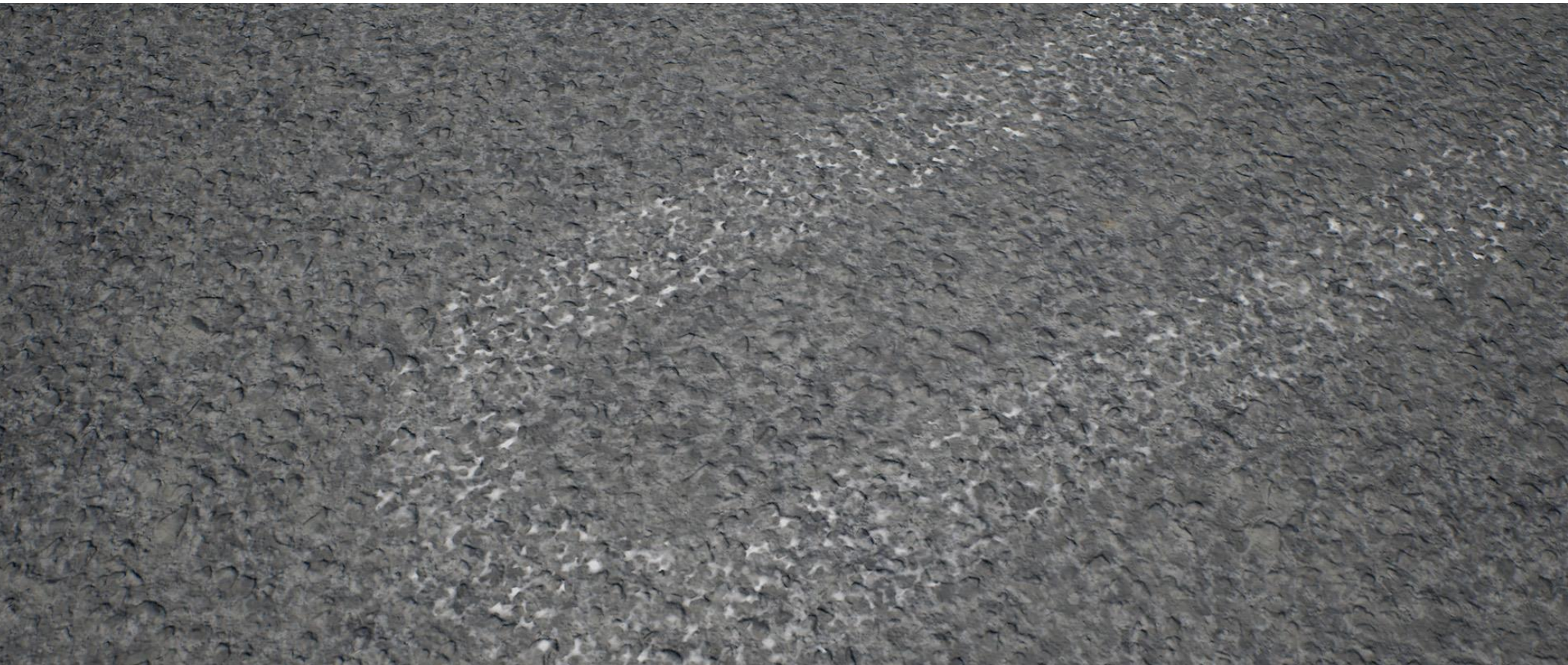
**白線かすれ表現** ⇒ 実物の白線を観察，劣化要因を3種に分類

**摩耗レベル小**　**塗膜がタイヤで削られ薄くなる**

# UEエディタをコンフィグレーターライクに利用

**白線かすれ表現** ⇒ 実物の白線を観察，劣化要因を3種に分類

**摩耗レベル大**　**アスファルトの凹みに塗料が残る**



# UEエディタをコンフィグレーターライクに利用


**白線かすれ表現**

要素をマテリアルパラメータ化,
シンプルな調整項目として設定

**強度/コントラスト/タイリング/回転**

# UEエディタをコンフィグレーターライクに利用

**頂点カラーペイントを使い要素毎に部分的な濃淡をつけられるように設定し，白線の多様なかすれ表現の作成に対応**

# UEエディタをコンフィグレーターライクに利用

**頂点カラーペイントを使い要素毎に部分的な濃淡をつけられるように設定し，白線の多様なかすれ表現の作成に対応**

# アノテーションデータ編

## 本開発でUEから出力するアノテーションデータ

- 駐車区画内のオブジェクト有無判定
- 駐車区画四隅の座標値
- 画像認識の教師データとして
  駐車車両ごとにID分けしたマスク画像

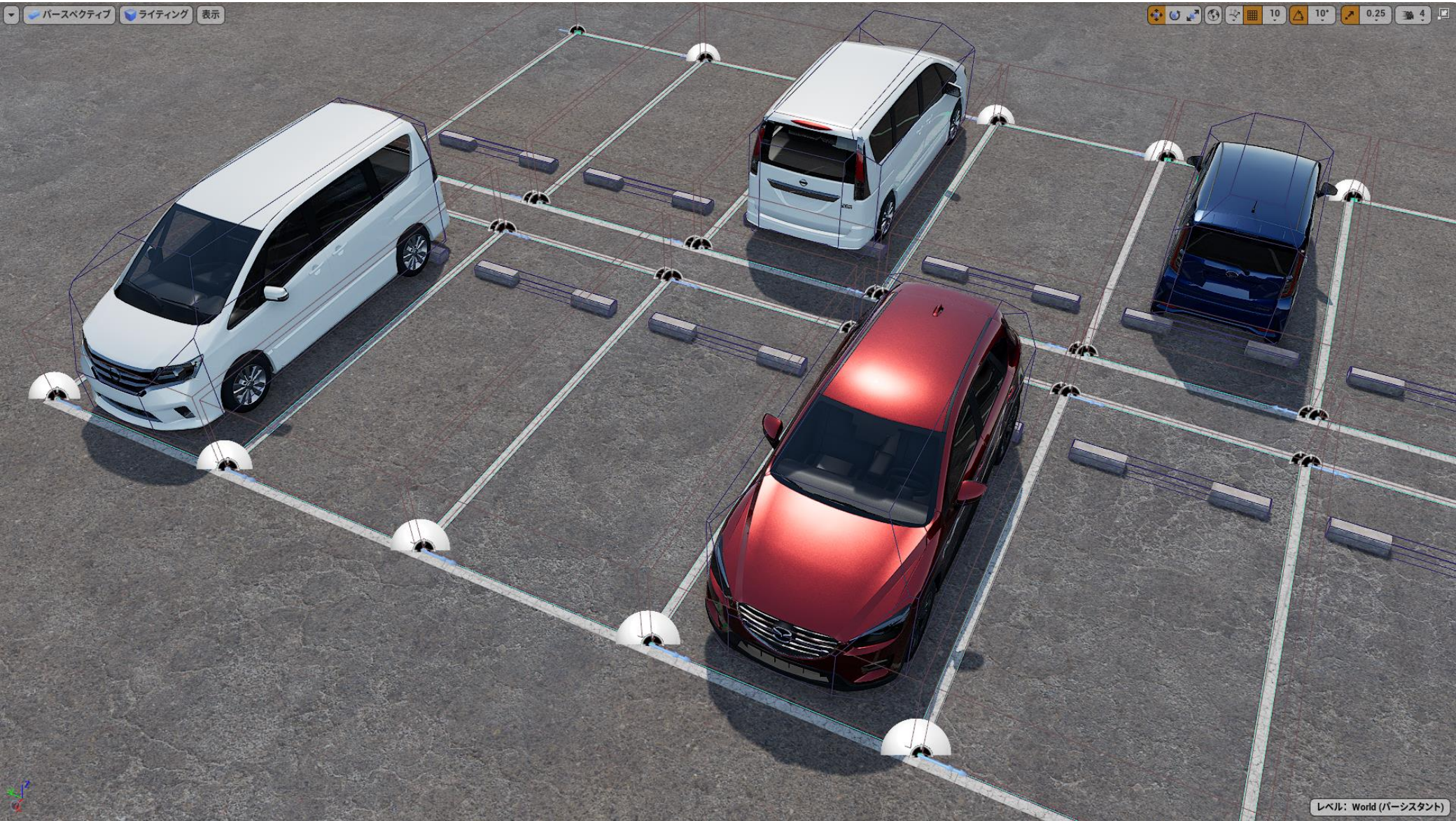
**上記情報を元に遮蔽判定や車両表示面積を算出**

# アノテーションデータ編

ゲームロジックと標準機能による
アノテーションに必要なデータの取得を試行

# ゲームロジックと標準機能を利用したデータ取得

**駐車区画内のオブジェクト有無判定** ⇒ コリジョン交差判定の利用



# ゲームロジックと標準機能を利用したデータ取得

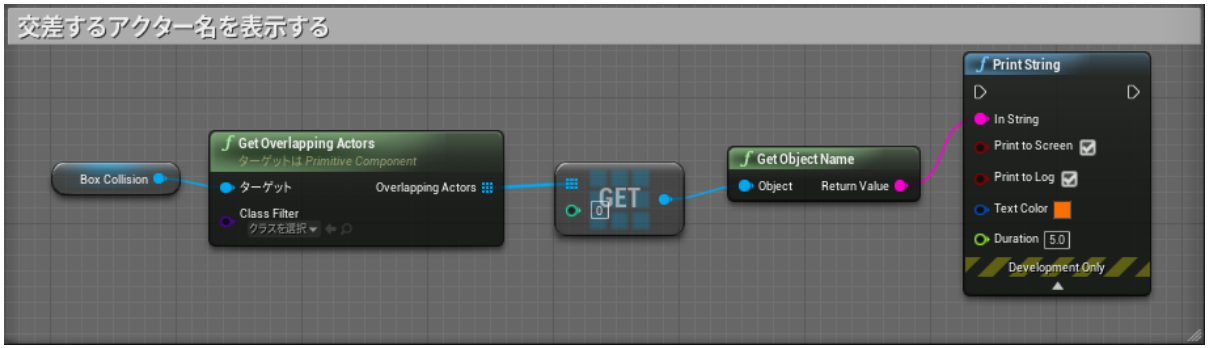
**駐車区画内のオブジェクト有無判定** ⇒ コリジョン交差判定の利用


**ゲーム内のイベントトリガーなどに用いたりする機能**

# ゲームロジックと標準機能を利用したデータ取得

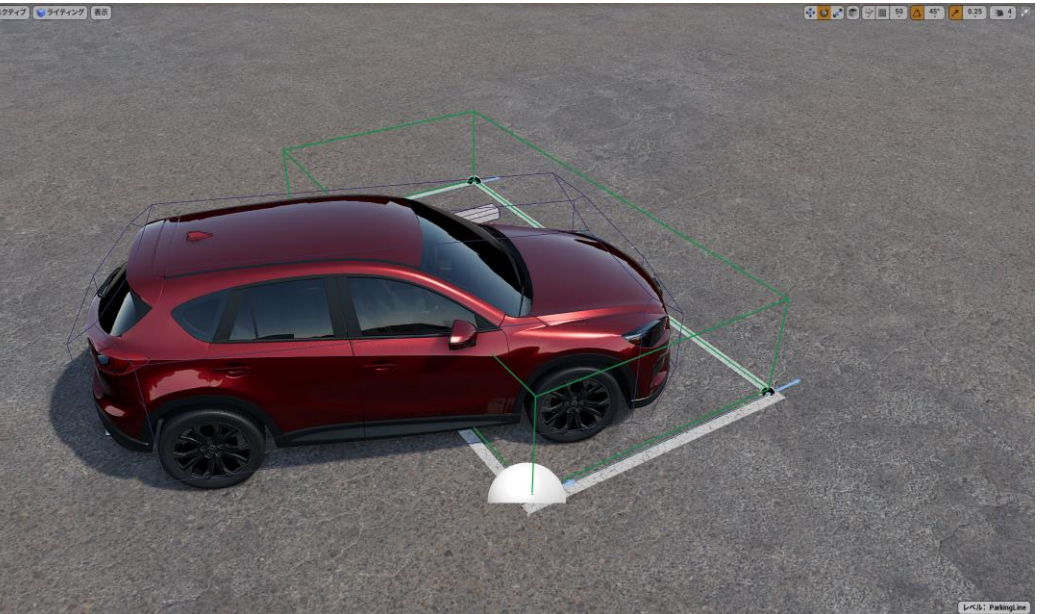
**駐車区画内のオブジェクト有無判定** ⇒ コリジョン交差判定の利用

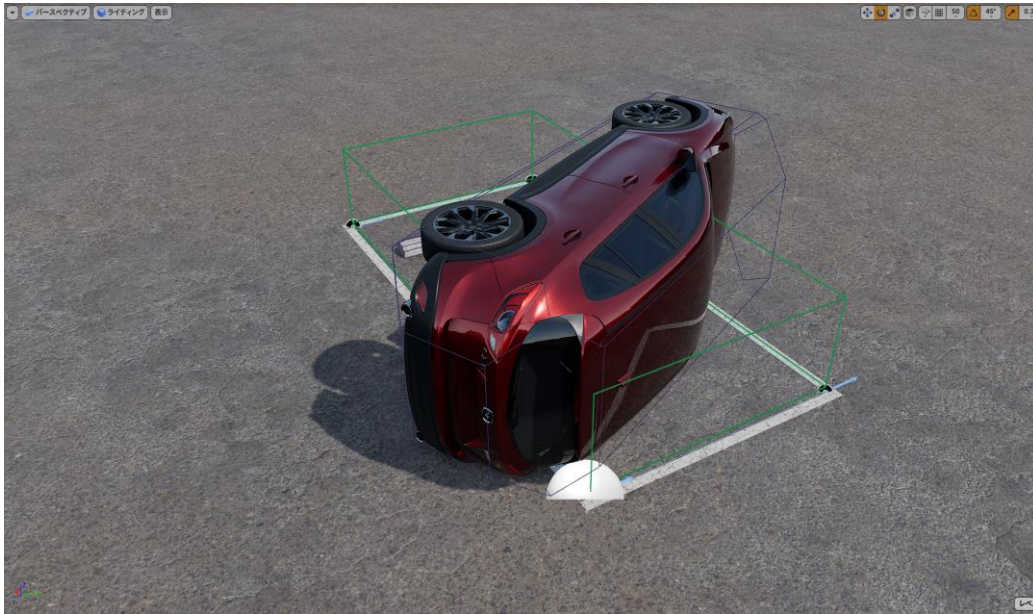
・容積のある物が領域内にあるかの判定が容易
・アクター名の取得可能




ログサンプル

```
[ParkingLine_003] CX5_004
[ParkingLine_006] Move_001
[ParkingLine_007] Serena_001
[ParkingLine_011] Serena_003
[ParkingLine_014] CX5_005
[ParkingLine_020] Move_005
[ParkingLine_021] CX5_006
[ParkingLine_022] Serena_011
```

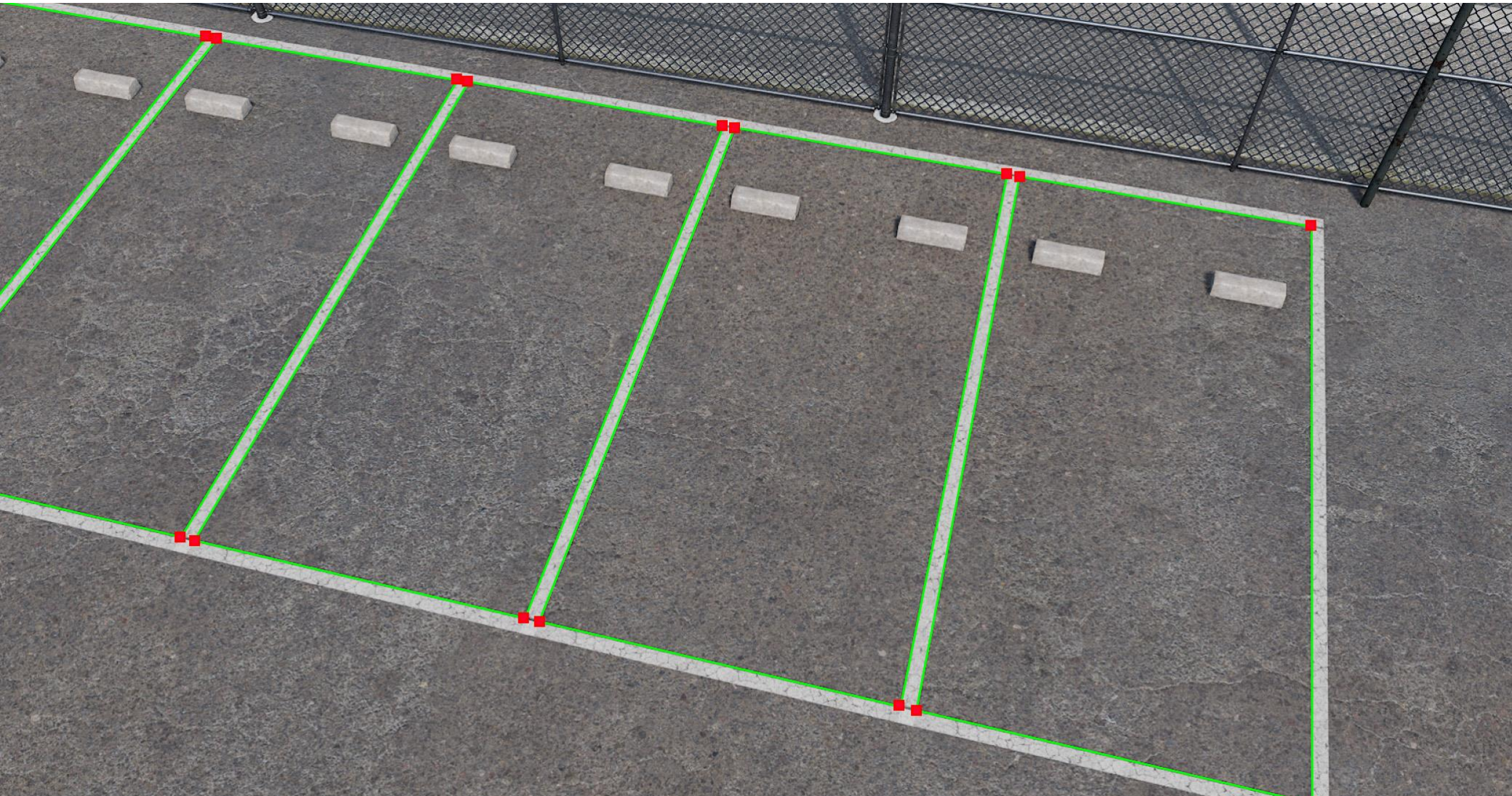




# ゲームロジックと標準機能を利用したデータ取得

**駐車区画四隅のスクリーン座標値検出**

➡ 区画四隅に配置したマーカーIDと座標値の取得



# ゲームロジックと標準機能を利用したデータ取得

**駐車区画四隅のスクリーン座標値検出**

➡ 区画四隅に配置したマーカーIDと座標値の取得

**ワールド座標をスクリーン座標に変換**
**⇒ キャラクター頭上にUIを表示する用途などで使用**

# ゲームロジックと標準機能を利用したデータ取得

## 駐車区画四隅のスクリーン座標値を検出

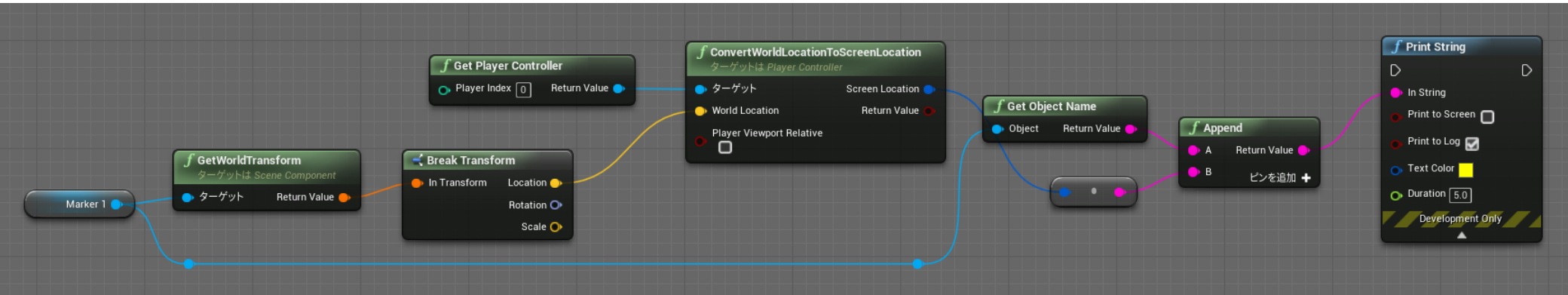
BPでTargetPointをマーカとして配置
コンポーネント名と座標値を毎フレーム取得


シミュレータによって
取得されたログサンプル

```
[ParkingLine_008] Marker1 X=546.251 Y=625.250
[ParkingLine_008] Marker2 X=791.973 Y=683.022
[ParkingLine_008] Marker3 X=939.391 Y=317.729
[ParkingLine_008] Marker4 X=749.562 Y=285.183

[ParkingLine_009] Marker1 X=802.657 Y=685.534
[ParkingLine_009] Marker2 X=1070.727 Y=748.560
[ParkingLine_009] Marker3 X=1150.130 Y=353.861
[ParkingLine_009] Marker4 X=947.558 Y=319.129

[ParkingLine_010] Marker1 X=1082.405 Y=751.306
[ParkingLine_010] Marker2 X=1376.017 Y=820.337
[ParkingLine_010] Marker3 X=1375.497 Y=392.500
[ParkingLine_010] Marker4 X=1158.854 Y=355.356
```



# ゲームロジックと標準機能を利用したデータ取得

## アノテーションIDマスク画像取得

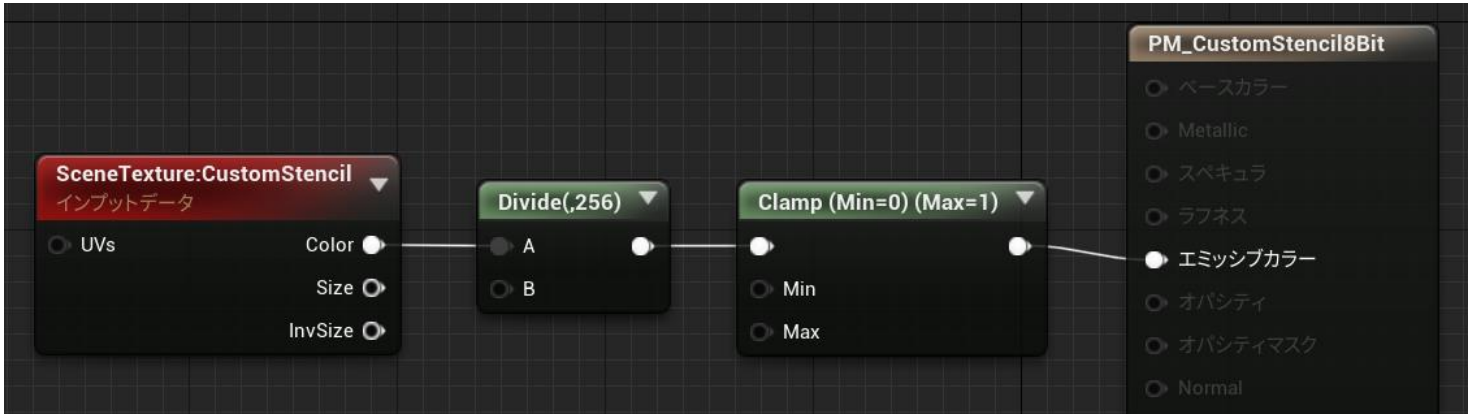
⇒ カスタムステンシルIDを利用し，色分けしたマスク画像を出力

# ゲームロジックと標準機能を利用したデータ取得

## アノテーションIDマスク画像取得

⇒ カスタムステンシルIDを利用し，色分けしたマスク画像を出力

車両ID番号をカラー値として出力するポストプロセスマテリアルを作成し,マスク画像からIDを取得




# 駐車場シミュレータ制作事例｜おわり

AISIN GROUP

Silicon Studio

# 疑似汚れ生成ツール開発

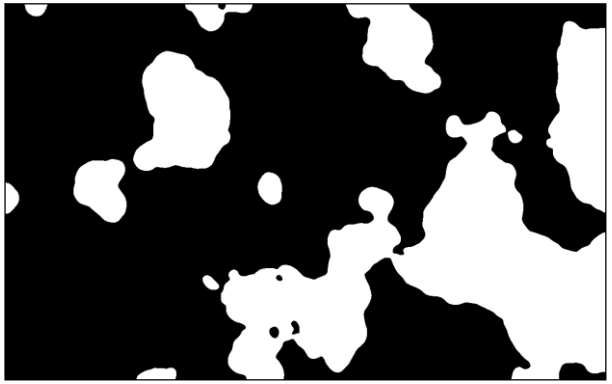
**アイシン単独(+他パートナー様)の試みとして泥/雨滴汚れの疑似生成ツールを開発**

・汚れ種類や汚れサイズ，占有率等のパラメータによる画像生成制御

入力画像

疑似汚れ付与画像

汚れ位置画像



**主要な汚れとして，下記3種類の汚れを定義**（画像は全てツールによる汚れ生成結果）

① 撥水性雨滴汚れ

② 親水性雨滴汚れ

③ 泥汚れ

# 疑似汚れ生成ツール｜効果検証

**学習データとしての疑似汚れの有用性を確認**
⇒ 検証結果詳細は ICPR2020 で発表予定

本物親水性雨滴汚れ　疑似親水性雨滴汚れ

本物泥汚れ　疑似泥汚れ

本物撥水性雨滴汚れ　疑似撥水性雨滴汚れ

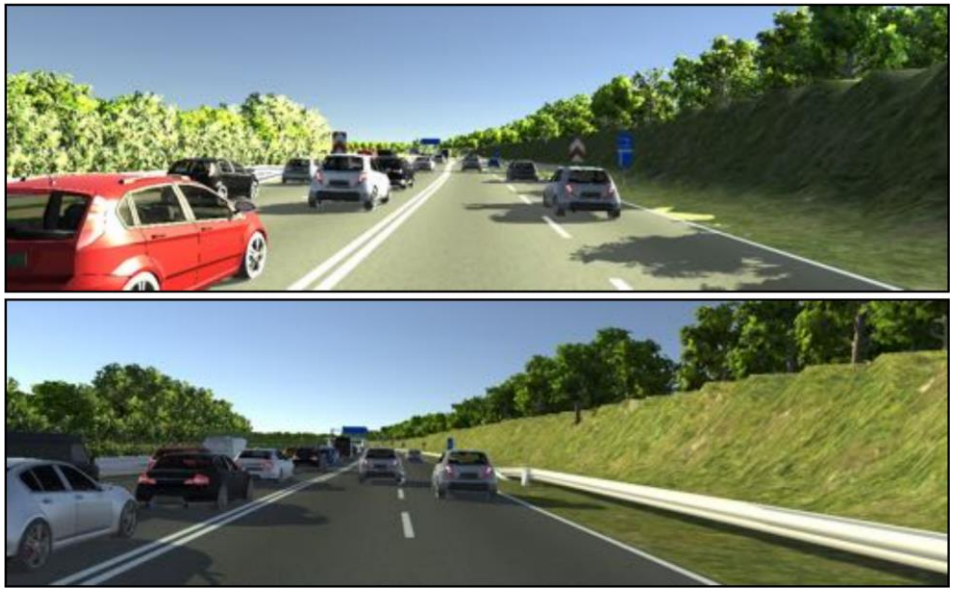
# 疑似汚れ生成ツール｜シミュレーションとの複合活用

## シミュレーション画像→実画像への画像変換技術などの活用を検討中

シミュレーション画像

学習に利用する
生成実画像

シミュレーション画像



生成画像

※引用：Learning Semantic Segmentation from Synthetic Data: A Geometrically Guided Input-Output Adaptation Approach, CVPR2019

# まとめ

**Unreal Engine 4 を利用して駐車場シミュレータを開発**

- ユーザーフレンドリーな操作
- 既存シミュレータでは表現できない多彩な駐車場シーンを再現可能

**かすれや埋没など多彩な変化を表現可能**

白線かすれ表現

トラロープ劣化表現

**疑似汚れ生成との併用による活用を検討**

汚れ無し画像

泥汚れ付与画像