

# VRM-NXで遊びながら学ぶ Pythonプログラミング

■ 角 卓

「Python」(パイソン)を使って「3D鉄道モデル」を動かす「鉄道模型シミュレーターNX」です。  
「プログラミング」と「鉄道」が好きな人にオススメです。

## 「VRM-NX」によるパターンダイヤ

今回は、「鉄道模型シミュレーターNX」(以下、「VRM-NX」)で、駅に一定時間停車する列車の自動運転を紹介しました。

今回は、編成を追加して、「特急列車」と「普通列車」による、追い越しのある自動運転を紹介します。

使用するレイアウトは、前回の駅が上下にある環状線(オーバル・レイアウト)を改造して、上側にポイントの待避線を追加します。

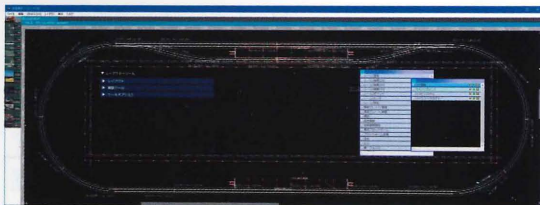


図1 待避線付きオーバル・レイアウト

「自動センサー」パーツ(以下、「センサ」)は、レイアウト左上と右下に、それぞれ、「上駅」「下駅」と名前を付けます。

分岐ポイントは「分岐A」「分岐B」の名前にします。

列車は、「特急列車」と「普通列車」の2編成を用意します。

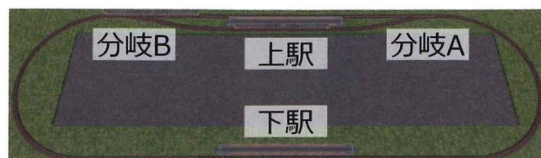


図2 ビュー表示

## パターン・ダイヤ

今回は、「特急列車」が「普通列車」を駅で追い越す、「緩急結合ダイヤ」を実現します。

周期的に繰り返すため、「パターン・ダイヤ」の一種になります。

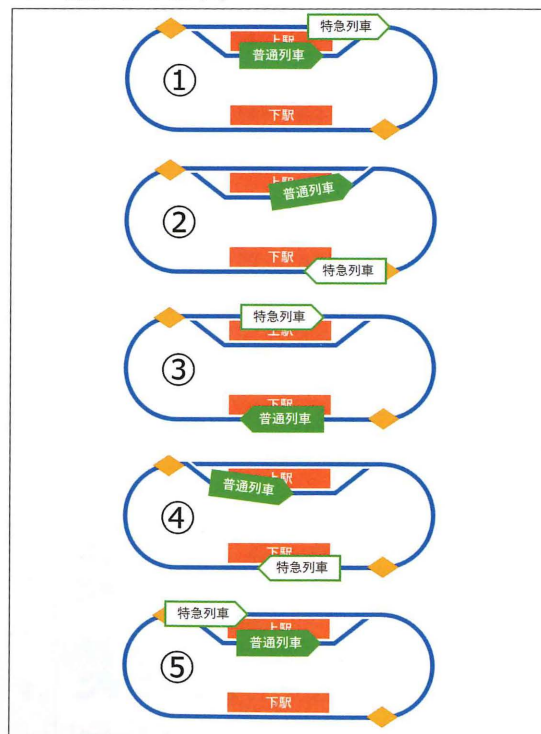


図3 パターンダイヤ

- ① 「特急列車」が「上駅」を先発。
- ② 「特急列車」が「下駅センサ」を踏んだら「分

岐A]を待避線側に切り替えて「普通列車」が発車。

「特急列車」は「下駅」を通過。

③「特急列車」が「上駅」に、「普通列車」は「下駅」に一定時間停車。

④「普通列車」は「上駅センサ」を踏むと、「分岐B」を待避線に切り替えて、「上駅」に停車。

⑤「特急列車」が「上駅センサ」を踏むと、「分岐B」「分岐A」のポイントを本線側に切り替えて、上駅に一定時間停車。

\*

⑥以降は、①に戻って繰り返す。

## スクリプト

関連するそれぞれの「スクリプト」を掲載します。

\*

先頭行の「def vrmevent\_xxx」は、レイアウト内の「パーツID」を示しています。

スクリプトをコピーするときは、「パーツID」と「スクリプト内のID」が一致しているか、確認してください。

### ■ 特急列車の「スクリプト」

特急列車の「スクリプト」には、「ビューー起動時」と、駅停車後のタイマーイベントで呼ばれる、「出発用速度制御」の「スクリプト」を記載します。

リスト 特急列車のスクリプト(抜粋)

```
def vrmevent_3(obj, ev, param):
    if ev == 'init':
        vrmapi.LOG(obj.GetName() + " 出発")
        obj.AutoSpeedCTRL(600.0, 0.75)
    elif ev == 'after':
        vrmapi.LOG(obj.GetName() + " 出発")
        obj.AutoSpeedCTRL(600.0, 0.75)
```

### ■ 普通列車の「スクリプト」

普通列車の「スクリプト」は、駅停車後のタイマーイベントで呼ばれる、「出発用速度制御」の「スクリプト」のみを記載します。

②のときとは別のセンサ・イベントが、直接、車両を出発させます。

リスト 普通列車のスクリプト(抜粋)

```
def vrmevent_42(obj, ev, param):
    if ev == 'init':
        dummy = 1
    elif ev == 'after':
        vrmapi.LOG(obj.GetName() + " 出発")
        obj.AutoSpeedCTRL(600.0, 0.75)
```

### ■ 「下駅」センサのスクリプト

「下駅」センサは、「特急列車」なら「下駅」を「通過」し、「普通列車」なら下駅に「停車」します。

「特急列車」と「普通列車」の識別は、センサの「GetTrain関数」で取得した列車オブジェクトの「GetName関数を使い、「名前が「特急列車」かどうか」の「if条件」で判断します。

また、「特急列車」の場合は、「execStart関数」で「普通列車」に対する「出発判定」を実施します。

リスト 「下駅」センサのスクリプト(抜粋)

```
#OBJID=149
import vrmapi
def vrmevent_149(obj, ev, param):
    if ev == 'init':
        dummy = 1
    elif ev == 'catch':
        # オブジェクト定義
        tr = obj.GetTrain()
        trn = tr.GetName()
        obn = obj.GetName()
        # 特急列車か
        if trn == '特急列車':
            # 特急列車通過処理
            vrmapi.LOG(trn+ " "+obn+"通過")
            l = vrmapi.LAYOUT()
            # 列車 [42] のセサ [118] 停車判定
            execStart(l.GetTrain(42), 118)
        else:
            # 普通列車停車処理
            vrmapi.LOG(trn+ " "+obn+"停車")
            tr.AutoSpeedCTRL(1050, 0.0)
            # 15秒後に出発
            tr.SetEventAfter(15.0)
```

### ■ 「上駅」センサのスクリプト

「上駅」センサは「列車名」を識別して「上駅」のポイント切り替えを行ないます。

「普通列車」は「分岐B」を「待避線」に切り替えて停車。

「特急列車」は「分岐A・B」両方を切り替えて、



「停車」と「出発タイマー」を定義します。

```
リスト 「上駅」センサのスクリプト(抜粋)

def vrmevent_118(obj, ev, param):
    if ev == 'init':
        dummy = 1
    elif ev == 'catch':
        # オブジェクト定義
        tr = obj.GetTrain()
        trn = tr.GetNAME()
        obn = obj.GetNAME()
        # ポイント定義
        l = vrmapi.LAYOUT()
        pl = l.GetPoint(51)
        :
```

## ■ レイアウトの「スクリプト」

「レイアウト・スクリプト」は、「起動時」に「普通列車」に「センサ情報」を登録します。

「setATS関数」は、列車オブジェクトがもつ連想配列に“section”文字列をキーとして、「上駅センサ・オブジェクト」を格納します。

これは「execStart関数」の実行時に、「上駅」に普通列車が存在しているかどうかを、引数の「パーツID」と「上駅センサ」の「GetID関数」での「値比較」で確認します。

さらに、「完全停車」後に出発させるため、「センサ確認」後に速度が「0」かどうかを確認しています。

\*

「出発処理」が完了したら、「連想配列」を「del」で削除します。

```
リスト レイアウトのスクリプト(抜粋)

def vrmevent(obj, ev, param):
    if ev == 'init':
        # 起動時初期登録
        sn = obj.GetATS(118)
        tr = obj.GetTrain(42)
        setATS(sn, tr)

# 列車の連想配列にセンサー obj を格納
def setATS(sn, tr):
    k = "section"
    d = tr.GetDict()
    d[k] = sn
```

```
trn = tr.GetNAME()
snn = d[k].GetNAME()
:
```

:

## 実行結果

「ビューワー」起動時のスクリプト「LOG ウィンドウ」のログは以下になります。

\*

センサ名を「上駅」「下駅」にしたことで、「センサ・イベント時の動作」が分かりやすくなります。

リスト 「スクリプトLOG」ウィンドウ

```
特急列車 出発
特急列車 下駅通過
分岐A[1]
普通列車 上駅出発
普通列車 下駅停車
分岐B[0]
分岐A[0]
特急列車 上駅停車
普通列車 出発
特急列車 出発
分岐B[1]
普通列車 上駅停車
特急列車 下駅通過
普通列車 未停車
分岐B[0]
分岐A[0]
特急列車 上駅停車
特急列車 出発
(繰り返し)
```

## 「鉄道模型シミュレーター」ならでは

今回のように、1つの本線に速度の異なる列車を走らせることは、現実の鉄道模型では実現難易度が高く、「鉄道模型シミュレーター」ならではの魅力の一つと言えるでしょう。

「鉄道模型シミュレーター NX」の仕様について最新情報を確認したい場合は、「VRM-NX SCRIPT MANUAL」(<https://vrmcloud.net/nx/script/>)で確認してください。



ソースコード全文、画像