シミュレーターの操作方法

Hibiki HATAKENAKA/畠中響生
2025/06/26

パス構成

```
- bin
        - itlts.exe
        - RecordExtraction.exe
- convert
        - no●●
                - vpos_no●●_r●●_ext.csv(出力ファイル)
        - run.bat
- Sentinel_LDK_Run-time_setup
        - Sentinel_LDK_Run-time_setup
                - HASPUserSetup.exe(トングル認識用)
- simcase
        - no●●(シナリオごと)
                - rand●●(乱数ごと)
                        - log
                        - vpos.csv(出力ファイル，車両軌跡)
                        - Case1_no●●_r●●_spd.csv(出力)
                        - Case1_no●●_r●●_vol.csv(出力)
                        - Case1_no●●_r●●_volspd.csv(出力)
                - Case1_no●●_r●●.mavn(ケースファイル)
                - run.bat
```

各ファイルの詳細

- 「simcase」内の「run.bat」
  - 各ケースファイルをitlts.exeが読み込んで実行

```
..\..\bin\itlts.exe -v -mode mav -csv rand01\Case1_no01_r01 -casepfx Case1_no01_r01 -caseext .mavn

..\..\bin\itlts.exe -v -mode mav -csv rand02\Case1_no01_r02 -casepfx Case1_no01_r02 -caseext .mavn

pause -1
```

各ファイルの詳細

- RecordExtraction.exeを実行しvpos.csvをvpos_no●●_r●●_ext.csvに書き換え
  - vpos_no●●_r●●_ext.csvの中身は不明
  - **全シナリオまとめて実行**

```
set nonum=01

..\bin\RecordExtraction.exe ../simcase/no%nonum%/rand01/ vpos.csv ./no%nonum%/ vpos_no%nonum%_r01_ext.csv

..\bin\RecordExtraction.exe ../simcase/no%nonum%/rand02/ vpos.csv ./no%nonum%/ vpos_no%nonum%_r02_ext.csv

..\bin\RecordExtraction.exe ../simcase/no%nonum%/rand03/ vpos.csv ./no%nonum%/ vpos_no%nonum%_r03_ext.csv

..\bin\RecordExtraction.exe ../simcase/no%nonum%/rand04/ vpos.csv ./no%nonum%/ vpos_no%nonum%_r04_ext.csv

..\bin\RecordExtraction.exe ../simcase/no%nonum%/rand05/ vpos.csv ./no%nonum%/ vpos_no%nonum%_r05_ext.csv

..\bin\RecordExtraction.exe ../simcase/no%nonum%/rand06/ vpos.csv ./no%nonum%/ vpos_no%nonum%_r06_ext.csv
```

書き換え用コード

```
// 01_小型自動レコード作成.cpp : このファイルには 'main' 関数が含まれています。プログラム実行の開始と終了がそこで行われます。
//

#include <iostream>
#include <string>
#include <vector>
#include "StringDivide.h"
#include "FileControl.h"
#define indat 10240
using namespace std;
```

書き換え用コード

```
int main(int argn, char *argv[])
{
    double rate = 0.1; //自動運転小型車比率変数
    string targetVehicleType;
    string readfn = "input/Case1_no1.mavn";
    string writefn = "output/Case1_no1.mavn";

    int count = 0;
    while (count < argn) {
        printf("%d, %s\n", count, argv[count]);
        count++;;
    }
```

書き換え用コード

```
//入出力ファイルは引数で設定
if (argn == 6) {
    rate = atof(argv[1]);
    int randSeed = atoi(argv[2]);
    targetVehicleType = argv[3];
    readfn = argv[4];
    writefn = argv[5];
    srand(randSeed);
    std::cout << "set Comammd Line  reate:" << rate <<  ", seed:" << randSeed << ", targetVehicleType:" << targetVehicleType <<  ", readfile:" << readfn << ", writefile:" << writefn << "\n";
}
else {
    std::cout << "need set Comammd Line 1:rate, 2:seed, 3:targetVehicleType, 4:readfile, 5:writefile\n";
    exit(0);
}
```

書き換え用コード

```
FILE* read = FileOpen(readfn, "r");
char buff[indat];

FILE *write = FileOpen(writefn, "w");

//string exchangeVehicleType = targetVehicleType + "_auto";

int n_nmlSum = 0;
int n_autSum = 0;

while (fgets(buff, indat, read) != NULL) {
    string buffStr = buff;//読み込んだレコードを保存
    vector<string> vecHeader;
    map<string, string> mapValue;

    //１行のレコードをキー(CLASS, ID, GENSCHEDULEなど)と値に分割してmapValueに格納
    //vecHeaderはキーを読み込んだ順番でリスト化
    GetValueSou(buff, mapValue,vecHeader);
```

書き換え用コード

```
if (mapValue["CLASS"] == "PacketGenerator") {
    if (mapValue["VEHICLETYPE"] == targetVehicleType) {
        map<int, string> mapValueNum;
        string numRec = mapValue["GENSCHEDULE"];
        sprintf_s(buff, sizeof(buff), "%s", numRec.c_str());
        GetValueInt2Str(buff, mapValueNum, ":");
        map<int, int>mapReGene;
        double rest = 0;
        int n_nml[50];//小型車の車両台数（整数値）
        int n_aut[50];//自動運転小型車の車両台数（整数値）
        for (int i = 0; i < 50; i++) {//6:30～19:00までの50個分のスケジュール配列を初期化
            n_nml[i] = 0;
            n_aut[i] = 0;
        }
        int sumaut = 0;
        for (int geneNum = 0; geneNum < mapValueNum.size(); geneNum++) {
            int num = atoi(mapValueNum[geneNum].c_str());
            double d_nml = num * (1 - rate);
            double d_aut = num * (rate);
            n_nml[geneNum] = d_nml;
            n_aut[geneNum] = d_aut;
            rest = d_nml - n_nml[geneNum];
```

書き換え用コード

```
            //printf("%d, %lf, %lf, %d, %d", num, d_nml, d_aut, n_nml[geneNum], n_aut[geneNum]);

            //端数の1台は小数点の割合で確率的に割り振る
            if (n_nml[geneNum] + n_aut[geneNum] != num) {
                double u = rand() / (double)RAND_MAX;
                if (u < rest) {
                    //printf(", %lf, %lf add nml", u, rest);
                    n_nml[geneNum]++;
                }
                else {
                    //printf(", %lf, %lf add auto", u, rest);
                    n_aut[geneNum]++;
                }
            }
            sumaut += n_aut[geneNum];
            //printf("\n");
        }
```

書き換え用コード

```
//検算用に合計値を足し合わせる
for (int i = 0; i < 50; i++) {
    n_nmlSum += n_nml[i];
    n_autSum += n_aut[i];
}

string rec = "";//ファイル出力レコード
int typeNumMax = 1;
if (sumaut != 0) typeNumMax = 2;
for (int typeNum = 0; typeNum < typeNumMax; typeNum++) {//小型車、自動運転小型車

    int count = 0;
    //レコードの順番を保持するためにvecHeaderのループを回す
    for (auto hd : vecHeader) {
        if (count != 0)rec +=",";
        if (hd == "ID") {
            if (typeNum == 0) {
                rec += hd + "=" + mapValue[hd];
            }
            else {
                rec += hd + "=" + mapValue[hd] + "_auto";//IDを自動運転小型車にする
            }
        }
```

書き換え用コード

```
else if (hd == "GENSCHEDULE") {
    rec += hd + "=" ;
    for (int geneNum = 0; geneNum < 50; geneNum++) {
        if (typeNum == 0) {
            rec += Int2Str(n_nml[geneNum]);
        }
        else {
            rec += Int2Str(n_aut[geneNum]);
        }
        if(geneNum != 49)rec += ":";
    }
}
else if (hd == "VEHICLETYPE") {
    if (typeNum == 0) {
        rec += hd + "=" + targetVehicleType;
    }
    else {
        rec += hd + "=" + targetVehicleType+"_auto";//車種を自動運転小型車にする
    }
}
```

書き換え用コード

```
                        else {
                            //指定のキー以外はそのまま出力
                            rec += hd + "=" + mapValue[hd];
                        }
                        count++;
                    }
                    rec += "\n";
                }
                fprintf(write, "%s", rec.c_str());
            }
            else {
                //対象の車種で無ければそのまま出力
                fprintf(write, "%s", buffStr.c_str());
            }
        }
        else {
            //PacketGeneratorクラスで無ければそのまま出力
            fprintf(write, "%s", buffStr.c_str());
        }
    }
    printf("result rate:%.2lf, normal:%d, auto:%d\n", n_autSum / (double)(n_nmlSum + n_autSum), n_nmlSum, n_autSum);
}
```

書き換え用コード

```
// プログラムの実行: Ctrl + F5 または [デバッグ] > [デバッグなしで開始] メニュー
// プログラムのデバッグ: F5 または [デバッグ] > [デバッグの開始] メニュー

// 作業を開始するためのヒント:
//    1. ソリューション エクスプローラー ウィンドウを使用してファイルを追加/管理します
//   2. チーム エクスプローラー ウィンドウを使用してソース管理に接続します
//   3. 出力ウィンドウを使用して、ビルド出力とその他のメッセージを表示します
//   4. エラー一覧ウィンドウを使用してエラーを表示します
//   5. [プロジェクト] > [新しい項目の追加] と移動して新しいコード ファイルを作成するか、[プロジェクト] > [既存の項目の追加] と移動して既存のコード ファイルをプロジェクトに追加します
//   6. 後ほどこのプロジェクトを再び開く場合、[ファイル] > [開く] > [プロジェクト] と移動して .sln ファイルを選択します
```

流れ

「書き換え用コード」の実行

- 「書き換え用コード」により手動車100%，施策無しのケースファイルを設定したいシナリオに合わせて書き換え
  - 「convert」のバッチファイルにも書き加える

課題

- 実行方法わからない
  - おそらくヘッダファイルがあれば実行できる
- 1ケースファイルに対して1ケースファイルになっている
  - ソースコードを書き換えればまとめて実行可能(?)
- 出力先がわからない
  - VSCode内にプロジェクトフォルダができて同じディレクトリに格納しておけば名前同じで中身が書き換えられる

流れ

simcase内のrun.batを実行

- itlts.exeが実行されケースファイルの条件下での交通流の結果が出力される

課題

- 時間短縮のためPC複数台で実行
- 各乱数ごとに並列化
- vpos.csvが非常にサイズが大きいので，rand01が終わったらすぐに圧縮，rand02が終わったらすぐに圧縮，，，という風にバッチファイルを書き換える必要がある

流れ

「convert」内のrun.batを実行

- RecordExtraction.exeが実行されvpos.csvをvpos_no●●_r●●_ext.csvに書き換え

課題

- ファイル名が変わるのはわかるが中身がどうなってるかはわからない
- もし先にrand01などを圧縮してしまった場合この操作をどうやればいいかわからない．全シナリオの出力ファイルに対して同時に操作することになっているが，vpos.csvを圧縮せずに全シナリオ同時に存在させるのはファイルの大きさ的に無理