Interface documentation for the VADERE software

Felix Dietrich

29th October 2014

Contents

1 Blackbox usage

VADERE can be used as a black box tool for microscopic pedestrian flow simulation. Fig. 1 shows the input - work - output scheme. A single file (called "scenario") is needed as input. One scenario input file generates one VADERE output file. The following text covers the input and output file specification in detail. Sample files are attached at the end.

Figure 1: Black box usage of VADERE software.

2 Input

VADERE can be run with a single file (called "scenario") as input. This file must contain a complete definition of a scenario, which is composed of several objects:

1. Simulation attributes
2. Pedestrian attributes
3. Model attributes
4. Topography
5. Output processors

The whole input file is in JSON format (http://json.org/), see section 'Sample files' below.

2.1 Simulation attributes

General attributes to configure the simulation:

1. simTimeStepLength: Simulated time per frame
2. realTimeSimTimeRatio: Ratio of simulated and real time in online visualization
3. writeSimulationData: Write simulation data? (yes:true / no:false)
4. digitsPerCoordinate: Accuracy of output data (digits per coordinate)
5. useRandomSeed, randomSeed: Random seed (useRandomSeed yes:true/no:false, randomSeed: integer)

2.2 Pedestrian attributes

At this time, only radius and parameters for the normally distributed free-flow speeds:

1. radius: Radius (pedestrians currently have circular shape in VADERE)
2. speedDistributionMean: Normal distribution mean
3. speedDistributionVariance: Normal distribution standard deviation
4. minimumSpeed: Minimum speed (cutoff, will choose again if lower)
5. maximumSpeed: Maximum speed (cutoff, will choose again if higher)
6. acceleration: Acceleration (not used by all models)

2.3 Model attributes

Possible microscopic models: Optimal Steps Model [Seitz and Köster(2012)] (which includes the possibility of a cellular automaton), Gradient Navigation Model [Dietrich and Köster(2014)] and Social Force Model [Helbing and Molnár(1995)]. Specific definitions of the parameters follow, listings 1 and 2 show how to setup the model section (see "attributesModel" in listing 8) so that the default parameters are used (empty braces are necessary here).

Listing 1: Default template for the parameters of the Gradient Navigation Model.

```
{
  "GRADIENT_NAVIGATION_MODEL": {},
  "FLOORFIELD": {},
  "PEDESTRIAN_POTENTIAL_GNM": {},
  "OBSTACLE_POTENTIAL_GNM": {}
}
```

Listing 2: Default template for the parameters of the Optimal Steps Model.

```
{
  "OPTIMAL_STEPS_MODEL": {},
  "FLOORFIELD": {},
  "POTENTIAL_COMPACT_SUPPORT": {}
}
```

2.4 Topography

A topography consists of the five following (lists of) objects:

1. Targets (of pedestrians)
2. Sources (of pedestrians)
3. Obstacles
4. Teleporter
5. Pedestrians present at the beginning of the simulation

as well as some metadata:

1. bounds (x,y,width,height): Dimension (height and width)
2. bounded: Boundary (true:yes / false:no)
3. boundingBoxWidth: Boundary width
4. finishTime: Time that VADERE has to simulate in total (simulated time, not runtime)

2.4.1 Shapes

In a topography, all extended objects (targets, sources, obstacles) have a "shape" (see example file, listing 7). Shapes can be polygons, circles and rectangles. When specifying a shape in JSON, it is important to add the "type" of the shape. The following listings contain examples of these three types of shapes. Polygons must be non-intersecting, but can be concave. Their points must be given in either clockwise or anti-clockwise order (in a normal cartesian coordinate system). Rectangles are defined by a point in their left bottom corner and their height and width.

Listing 3: An example of a polygon.

```
"shape": {
    "type": "POLYGON",
    "points": [
```

```
      {
        "x": 15.1,
        "y": 1.7
      },
      {
        "x": 15.0,
        "y": 3.9
      },
      {
        "x": 16.9,
        "y": 4.2
      }
    ]
}
```

Listing 4: An example of a circle.

```
"shape": {
    "radius": 0.4,
    "center": {
      "x": 16.7,
      "y": 5.9
    },
    "type": "CIRCLE"
}
```

Listing 5: An example of a rectangle.

```
"shape": {
    "x": 20.0,
    "y": 5.3,
    "width": 4.0,
    "height": 5.0,
    "type": "RECTANGLE"
}
```

2.4.2 Targets

Targets define regions where pedestrians want to go. The pedestrian dynamics model used for the simulation uses this information to steer the pedestrians. A target has the following properties:

- a "shape" (see section 2.4.1).
- an "id" (integer), which is an identifier for this specific target.
- "absorbing" (boolean), states whether the target should remove pedestrians with this region set as target if they step onto it.

2.4.3 Sources

Sources define regions where pedestrians are created (and start walking). A source has the following properties:

- a "shape" (see section 2.4.1).
- an "id" (integer), which is an identifier for this specific source.
- "spawnNumber" (integer), specifies how many pedestrians should be created in each creation step.
- "spawnDelay" (float), specifies how many seconds this source waits between two creation steps.

- "startTime" (float), specifies when the source should start creating pedestrians.
- "endTime" (float), specifies when the source should stop creating pedestrians. If "startTime" and "endTime" coincide, the source only creates pedestrians once.
- "spawnAtRandomPositions" (boolean), specifies whether the pedestrians should be created randomly (true) in the source region or equally spaced (false).
- "useFreeSpaceOnly" (boolean), if enabled (true), the source creates pedestrians only when there is enough free space in the source region. If this is enabled and a pedestrian cannot be created, it is created as soon as there is space available.

2.4.4 Initial Pedestrians

The list "pedestrians" in the topography (see example listing 7) may contain single pedestrians that are present from start at specific locations. These pedestrians can only have a "position" and a list of "targetIds".

2.5 Output processors

One can specify how the output of VADERE should be processed after the simulation. This is possible via 'output processors', which can be specified directly in the input file. There are several output processors available, a detailed explanation of them will follow shortly. The output is always formatted in tables, an example is given in listing 6.

Listing 6: A sample output of trajectories in VADERE.

```
id time x y
1 0.000000 3.425873 6.565516
2 0.000000 9.479350 1.572532
3 0.000000 8.781264 3.460840
4 0.000000 2.151976 3.920817
5 0.000000 2.132316 6.418738
1 0.500000 3.425873 6.565516
2 0.500000 9.479350 1.572532
3 0.500000 8.781264 3.460840
4 0.500000 2.151976 3.920817
5 0.500000 2.132316 6.418738
```

3 Output

One scenario input file generates one VADERE output file, which is composed of four sections:

1. Metadata
2. Input file
3. Pedestrian trajectories (deprecated)
4. Pedestrian targets (deprecated)

In the first two sections, the data is stored in JSON format (http://json.org/). The last two sections use a special format, and are deprecated since the "output processors" can produce trajectories and target assignments (explained in 2.5).

3.1 Metadata

The metadata part of an output file contains three SHA-1 hashes:

1. attributes hash
2. topography hash
3. commit hash

The hashes can be used to efficiently compare the contents of input files when searching for specific output files (and vice versa). "commit hash" means the hash of the latest GIT commit of the Software repository, to be able to use the same exact software when trying to reproduce a given result.

3.2 Input file

This part contains the exact same data that was present in the input file generating this output. It can be used to verify that a given output file was indeed created using a given input file (i.e. with the same parameters and topography).

3.3 Pedestrian trajectories (deprecated)

Pedestrian trajectories are represented by a list of time steps, containing the individuals positions in the following format:

@stepID1 TIME1
ID1 X1 Y1 TYPE1
ID2 X2 Y2 TYPE2
... ...
@stepID2 TIME2
ID1 X1 Y1 TYPE1
ID2 X2 Y2 TYPE2
... ...

where `TYPE#` is one of `{c, m, d}` (created, moved, deleted). Note that this type of output is deprecated and is replaced by separate files in the future (see listing 2.5).

3.4 Pedestrian targets (deprecated)

The trajectories of pedestrians do not directly reveal which target they try to reach. Therefore, this section stores a many-to-one map (or many-to-many in case of multiple targets per ped.) of pedestrian IDs and target IDs in the following format:

ID1 TARGETID-of-ped-ID1
ID2 TARGETID-of-ped-ID2
... ...

Note that this type of output is deprecated and is replaced by separate files in the future (see listing 2.5).

4 Example files

Listing 7: A sample input file for VADERE.

```
{
  "vadere": {
    "attributesModel": {
      "GRADIENT_NAVIGATION_MODEL": {...},
      "FLOORFIELD": {...},
      "PEDESTRIAN_POTENTIAL_GNM": {...},
      "OBSTACLE_POTENTIAL_GNM": {...}
    },
    "attributesPedestrian": {
      "radius": 0.195,
      "speedDistributionMean": 1.34,
      "speedDistributionVariance": 0.26,
      "minimumSpeed": 0.3,
      "maximumSpeed": 3.0,
      "acceleration": 2.0
    },
    "attributesSimulation": {
      "simTimeStepLength": 0.25,
      "realTimeSimTimeRatio": 0.0,
      "writeSimulationData": true,
      "digitsPerCoordinate": 2,
      "useRandomSeed": false,
      "randomSeed": 1
    },
    "topography": {
      "attributes": {
        "finishTime": 300.0,
        "bounds": {"width": 30.0,
          "height": 9.0
        },
        "boundingBoxWidth": 0.5,
        "bounded": true
      },
      "obstacles": [
        {
          "shape": {
            "type": "POLYGON",
            "points": [
              {
                "x": 30.0001,
                "y": 8.4999
              },
              {
                "x": 30.0001,
                "y": 9.0001
              },
              {
                "x": -1.0E-4,
                "y": 9.0001
              },
              {
                "x": -1.0E-4,
                "y": 8.4999
              }
            ]
          },
          "id": -1
```

```
      }],
    "targets": [
      {
        "id": 1,
        "absorbing": true,
        "shape": {
          "x": 28.0,
          "y": 1.0,
          "width": 1.0,
          "height": 7.0,
          "type": "RECTANGLE"
        }
      }
    ],
    "sources": [
      {
        "id": -1,
        "shape": {
          "x": 1.0,
          "y": 1.0,
          "width": 9.0,
          "height": 7.0,
          "type": "RECTANGLE"
        },
        "spawnDelay": 1.0,
        "spawnNumber": 180,
        "startTime": 0.0,
        "endTime": 0.0,
        "spawnAtRandomPositions": true,
        "useFreeSpaceOnly": false,
        "targetIds": [
          1
        ]
      }
    ],
    "pedestrians": [
                        {
                          "position": {
                                "x": 13.48,
                                "y": 1.0
                          },
                          "targetIds": [
                                1
                          ]
                        },
                        {
                          "position": {
                                "x": 13.870000000000001,
                                "y": 1.6754998149518623
                          },
                          "targetIds": [
                                1
                          ]
                        }
                ]
  }
},
"name": "inputfile",
```

```
  "outputProcessors": [...]
}
```

Listing 8: A sample output file for VADERE.

```
$ metadata
{
  "attributeshash": "...",
  "topographyhash": "...",
  "commithash": "..."
}

$ topography
{
  "attributes": {
    "finishTime": 300.0,
    "bounds": {
      "x": 0.0,
      "y": 0.0,
      "width": 30.0,
      "height": 9.0
    },
    "boundingBoxWidth": 0.5,
    "bounded": true
  },
  "obstacles": [
        {
          "shape": {
                "type": "POLYGON",
                "points": [
                  {
                        "x": 30.0001,
                        "y": 8.4999
                  },
                  {
                        "x": 30.0001,
                        "y": 9.0001
                  },
                  {
                        "x": -1.0E-4,
                        "y": 9.0001
                  },
                  {
                        "x": -1.0E-4,
                        "y": 8.4999
                  }
                ]
          },
          "id": -1
        }
        ],
  "targets": [
        {
          "id": 1,
          "absorbing": true,
          "shape": {
                "x": 28.0,
                "y": 1.0,
```

```
                "width": 1.0,
                "height": 7.0,
                "type": "RECTANGLE"
          }
        }
  ],
  "sources": [
        {
          "id": -1,
          "shape": {
                "x": 1.0,
                "y": 1.0,
                "width": 9.0,
                "height": 7.0,
                "type": "RECTANGLE"
          },
          "spawnDelay": 1.0,
          "spawnNumber": 180,
          "startTime": 0.0,
          "endTime": 0.0,
          "spawnAtRandomPositions": true,
          "useFreeSpaceOnly": false,
          "targetIds": [
                1
          ]
        }
  ],
  "pedestrians": []
}

$ attributes
{
  "attributesPedestrian": {
          "radius": 0.195,
          "speedDistributionMean": 1.34,
          "speedDistributionVariance": 0.26,
          "minimumSpeed": 0.3,
          "maximumSpeed": 3.0,
          "acceleration": 2.0
  },
  "attributesSimulation": {
      "simTimeStepLength": 0.25,
      "realTimeSimTimeRatio": 0.0,
      "writeSimulationData": true,
      "digitsPerCoordinate": 2,
      "useRandomSeed": false,
      "randomSeed": 1
  },
  "attributesModel": {
    "GRADIENT_NAVIGATION_MODEL": {...},
    "FLOORFIELD": {...},
    "PEDESTRIAN_POTENTIAL_GNM": {...},
    "OBSTACLE_POTENTIAL_GNM": {...}
  }
}

$ trajectories
@1 0.00
```

```
1 3.24 4.72 c
2 9.41 4.62 c
3 2.39 6.97 c
...
1 3.24 4.72 m
2 9.41 4.62 m
3 2.39 6.97 m
...

@2 0.25
1 3.24 4.72 m
2 9.45 4.66 m
3 2.40 6.97 m
...


$ pedestrianData
1 1
2 1
3 1
...
```

References

[Dietrich and Köster(2014)] Felix Dietrich and Gerta Köster. Gradient navigation model for pedestrian dynamics. *Physical Review E*, 89:062801, 2014. doi: 10.1103/PhysRevE.89.062801.

[Helbing and Molnár(1995)] Dirk Helbing and Péter Molnár. Social Force Model for pedestrian dynamics. *Physical Review E*, 51(5):4282–4286, 1995. doi: 10.1103/PhysRevE.51.4282.

[Seitz and Köster(2012)] Michael J. Seitz and Gerta Köster. Natural discretization of pedestrian movement in continuous space. *Physical Review E*, 86:046108, 2012. doi: 10.1103/PhysRevE.86.046108.