Vecto 3.x Release Notes

DI Dr. Markus Quaritsch, DI Dr. Christian Kreiner, Michael Krisper

November 12, 2015

1 Vecto 3.0.1 Release Notes

1.1 General Notes

Vecto 3.0 has been rewritten from scratch. It is now a dynamic program library that contains the core simulation and can be embedded in other applications.

Vecto 3.0.1 has been integrated in the graphical user interface of Vecto 2.2 via an additional "Start" button on the main screen (cf. Figure 1). Additionally, a command-line program is provided to run multiple Vecto jobs (cf. Figure 2). For more information how to use the command-line program please see "vectocmd.exe - h".

In case you find a bug or Vecto 3.0 does not behave as expected **please follow the instructions** given in Section 2.

1.2 Changes in Vecto 3.0.1

- New distance-based simulation model

- Simulation with variable steps. The simulation distance is adapted such that the respective time is about $0.5\,\mathrm{s}$ (parameter TargetTimeInterval). Certain simulation events are calculated using adapted time intervals (e.g. for traction interruption or to exactly meet the target speed vs. distance cycle).

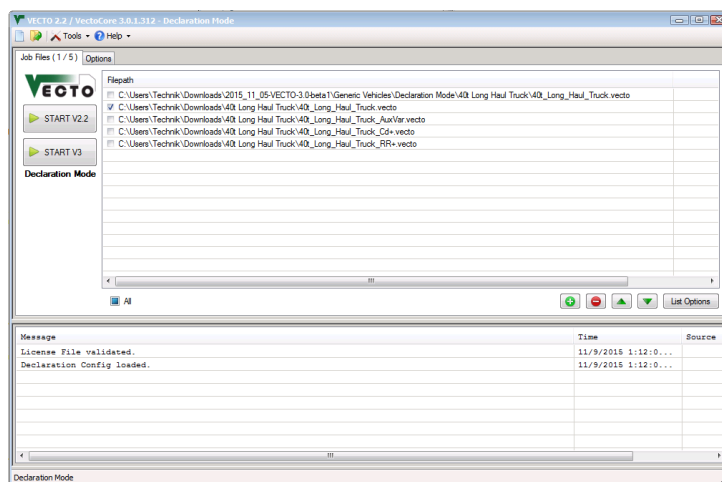- Accuracy of simulated distance vs. driving cycle distance $< 1\,\mu\mathrm{m}$

Figure 1: Vecto 3.0.1 Graphical User Interface

Figure 2: Vecto 3.0.1 Command Line Program

- New component-based software architecture
  - Separate software component for every part of the power-train: Combustion Engine, Clutch, Gearbox, Retarder, Axle Gear Vehicle, Wheels, Driver, Driving Cycle

    The models of every component are taken from Vecto 2.2
  - Interfaces for components reflecting physical quantities: force/velocity, angular speed/torque, ...
  - Power-train components modularly usable, custom power-train configurations possible

- DRIVERSTRATEGY as separate component with well-defined interface

- GEARSHIFTSTRATEGY as separate component with well-defined interface

- All computations are done in SI units; data-types reflect concrete SI units (i.e., kg, m, s, Nm, ...)

- Run-time conformity check of SI units on all computations and SI conversions. (might be relaxed for final relase version)

- Traction interruption of arbitrary time possible (also $< 1\,\mathrm{s}$), accurate simulation of traction interruption time

- New IDLE-CONTROLLER to simulate engine speed during traction interruption periods

- Intelligent search of operating point during coasting, braking, and in engine-overload situations, deviation from FLD $< 0.5\,\mathrm{W}$ (parameter ENGINEPOWERSEARCHTOLERANCE).

- Exact analytic solutions instead of exhaustive search where possible, e.g., computing engine's preferred speed, computing time interval required for driving a certain distance, compute distance required to decelerate to a given target speed depending on the driver's deceleration curve, computation of distance to drive when accelerating before braking, ...

- Support sparse representation of driving cycles. Instead of specifying the driving cycle on a per-meter basis only distance entries with speed or slope changes need to be specified. Reduction of 96 % in disk space

- Multi-threaded execution when multiple driving cycles resp. multiple Vecto jobs are executed

- Vecto 3.0.1 only supports the latest version of the input file formats (JSON), i.e., .vecto: v 2; .vveh: v 7; .vgbx: v 5; .veng: v 3

Name	Not Covered	Not Covered %	Covered	Covered %
Technik_FITIPC 2015-11-09 16_29_02.coverage	2277	11,86%	16929	88,14%
vectocore.dll	1684	15,31%	9318	84,69%
TUGraz.VectoCore.FileIO.EngineeringFile	4	40,00%	6	60,00%
TUGraz.VectoCore.Models.SimulationComponent.Data.Engine	170	35,42%	310	64,58%
TUGraz.VectoCore.Models.SimulationComponent.Data.Gearbox	74	34,26%	142	65,74%
TUGraz.VectoCore.Models.SimulationComponent.Data	292	30,39%	669	69,61%
TUGraz.VectoCore.FileIO	12	24,00%	38	76,00%
TUGraz.VectoCore.Exceptions	10	20,83%	38	79,17%
TUGraz.VectoCore.Utils	168	17,07%	816	82,93%
TUGraz.VectoCore.FileIO.Reader.Impl	68	15,81%	362	84,19%
TUGraz.VectoCore.Models.Simulation.Impl	96	15,12%	539	84,88%
TUGraz.VectoCore.Models.SimulationComponent.Impl	542	14,82%	3116	85,18%
TUGraz.VectoCore.Models.Simulation.Data	96	12,47%	674	87,53%
TUGraz.VectoCore.FileIO.Reader	50	11,79%	374	88,21%
TUGraz.VectoCore.FileIO.Reader.DataObjectAdaper	44	9,78%	406	90,22%
TUGraz.VectoCore.Models.Declaration	58	3,17%	1769	96,83%
Global Classes	0	0,00%	10	100,00%
TUGraz.VectoCore.Configuration	0	0,00%	13	100,00%
TUGraz.VectoCore.FileIO.DeclarationFile	0	0,00%	5	100,00%
TUGraz.VectoCore.Models	0	0,00%	10	100,00%
TUGraz.VectoCore.Models.Connector.Ports.Impl	0	0,00%	14	100,00%
TUGraz.VectoCore.Models.SimulationComponent	0	0,00%	7	100,00%

Figure 3: Screenshot of Test-coverage Report (Microsoft Visual Studio)

- Support for arbitrary order of columns in CSV files if the header is given as specified in the Manual. If the header is not recognized: fall-back to column-based parsing

- more energy-efficient implementation of *Overspeed*: vehicle acceleration only due to road's slope, engine is on drag-curve

- Use of well-known logging framework (NLog) allows to configure logging output based on severity and namespace. Supports different logging targets

- Implemented in C#, requires .Net Framework version 4.5

- Use of latest libraries for logging, parsing JSON files, and processing of PDF files

- 270 unit- and integration-tests developed; 262 tests passed by Vecto release; 4 tests failed due to unreasonable vehicle configuration (torque outside loss-map range), 4 tests failed due to unreasonable driving cycles (15 % and 25 % uphill)
  - Code coverage: 84 %
    The main parts of the code are covered by tests, parts with low coverage are mainly auto-generated equality methods and parsing of input data (cf. Figure 3)

- Code complexity[1]: currently the max. complexity is 20 (single method) and 10 methods have a complexity >10. The most complex methods are in the ShiftStrategy, the DriverStrategy, and the Powertrain builder; see Figure 4.

1.3 Open Issues

- Sanity checks on input data

- Refactoring of DefaultDriverStrategy to handle certain situations when braking is required during coasting to meet small target-speed changes during coasting.

- Status output to GUI/Console during simulation

- Advanced Driver Assist Systems

- Adaptation of component models to most recent ACEA Whitebook

---

[1]According to *SourceMonitor*

Figure 4: Screenshots of code metrics (SourceMonitor): Checkpoints over time (top), list of classes sorted by complexity (middle), List of methods sorted by complexity (bottom)

2 How to Submit a Bug Report

In case you encounter a bug or Vecto 3 is not behaving as you would expect it is of utmost importance that the developers can reproduce your results[2]. All bugs should be submitted as Jira Issue (either *Bug* or *Feature/Task*) to maintain traceability.

Every bug-report should contain a detailed description containing:

- What did you do

- What is your expected outcome

- What is the actual outcome

- Which Vecto version did you use? (Version number and Build number)
  The version number is printed when you start vectocmd in verbose mode

- Screenshots if necessary

In addition a detailed log of your simulation is necessary. Please follow these instruction to create a log-file and the mod-files:

- **Start a command-line**
  Press *Windows-r* and type `cmd`, change to the directory where you extracted Vecto 3.0

- Delete old log-files:
  `del logs\*`

- Start the simulation using Vecto-Commandline:
  `vectocmd.exe -t -vvvv -mod <jobfile.vecto>`
  it is important to use the option `-vvvv` and `-mod` which enable verbose logging and output of the mod-data file (`.vmod`.

  If logging is enabled, the simulation runs are executed sequentially. If the error occurs only in some simulation runs (i.e., driving cycle and loading) you can cancel the other simulation runs by pressing *Ctrl-c*. (Note: if verbose logging is enabled the simulation takes much longer)

- File a new Jira issue[3] with a detailed description (see above)

- Attach all data necessary to reproduce your results to the Jira issue, or if the data is confidential you can directly send it to the developers.
  - All input files
  - All Log-files in the directory `logs` (compressed)
  - Mod-files (stored in the directory of the Vecto job)

---

[2] http://www.chiark.greenend.org.uk/~sgtatham/bugs.html
[3] https://webgate.ec.europa.eu/CITnet/jira/secure/CreateIssue!default.jspa