

# Fast generic sensor models for testing highly automated vehicles in simulation

M. Stolz, G. Nestlinger

Automated driving is one of the big trends in automotive industry nowadays. Over the past decade car manufacturers have increased step by step the number and features of their driving assistance systems. Besides technical aspects of realizing a highly automated vehicle, the validation and approval poses a big challenge to automotive industry. Reasons for this are grounded in the complexity of interaction with the environment, and the difficulties in proving that an automated vehicle shows at least the same performance as a human driver, with respect to safety. There is a common agreement within the community, that physical vehicle testing only, will practically not be feasible for validation. Simulation is seen as a key enabler for validation as well as for development of highly automated vehicles. Several commercial simulation tools are available to model and test the entire chain of “sens – plan – act”. But still there is a lack of fast generic sensor models, for the use from development to vehicle in the loop tests. This article proposes a fast generic sensor model to be used in testing and validation of highly automated driving functions in simulation.

Keywords: sensor models; automated driving; radial basis functions

## **Schnelle generische Sensormodelle für das Testen hochautomatisierter Fahrzeuge in der Simulation.**

*Automatisiertes Fahren ist heutzutage das zentrale Thema in der Automobilindustrie. Über die letzten Dekaden hinweg haben Autohersteller schrittweise die Anzahl sowie den Funktionsumfang von Fahrassistenzfunktionen in ihren Fahrzeugen erhöht. Abgesehen von technischen Aspekten bei der Umsetzung hochautomatisierter Fahrzeuge, stellen deren Validierung und Absicherung aufgrund der Komplexität der Interaktion mit dem Umfeld eine große Herausforderung dar. Es gibt allgemeine Übereinstimmung darüber, dass ausschließlich physikalische Absicherung kein praktisch durchführbarer Ansatz ist. Als ein möglicher Lösungsweg bietet sich die Simulation an. Hierbei steht eine Vielzahl von kommerziellen Tools zur Modellierung und Validierung der kompletten Wirkkette “Wahrnehmen – Planen – Agieren” zur Verfügung. Trotz dieser Fortschritte mangelt es immer noch an schnellen generischen Sensormodellen. Dieser Beitrag stellt ein generisches Sensormodell für das simulative Testen und Validieren von hochautomatisierten Fahrfunktionen vor.*

*Schlüsselwörter: Sensormodelle; automatisiertes Fahren; radiale Basisfunktion*

Received March 23, 2018, accepted May 24, 2018, published online July 16, 2018  
© The Author(s) 2018



## 1. Introduction

Validation of highly automated vehicles (HAVs) is practically infeasible by applying conventional testing methods like real test drives: In [13, Ch. 63.4.1] it was pointed out, that 240 000 000km of test drives are required for an autonomous vehicle resulting in half the number of accidents involving personal injury compared to human-guided vehicles.

Simulation is a very promising approach to validate HAVs, as it enables e.g.:

- testing of safety critical situations,
- involving the environment into tests and
- perform accelerated testing (faster than real time).

During development and validation, software-frameworks are used to model and co-simulate the static environment (road, static obstacles, buildings, etc.), other dynamic traffic participants (other vehicles, pedestrians, animals, etc.), and the automated vehicle under test, as well as their interaction [12]. There are several commercial and open source software packages available which are suitable for the simulation of HAVs within their surrounding (e. g. [1, 2, 7–9, 11]).

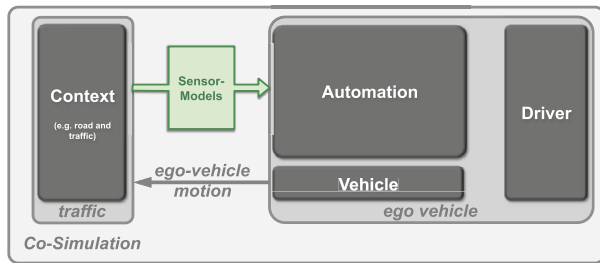
Despite public approaches like [5], up to now there is no common agreement on the interface linking software for environment simulation and the perception part of vehicle-software, responsible

for automated driving. At present, a common approach of most environment simulation tools is to use some kind of *object-list* transferring the *ground truth* regarding objects of the environment simulation to the sensor models. Note that real sensors (e.g. cameras and radar) equipped with specific post-processing of the sensor raw data in most cases also use some kind of *object-list* as interface to the software responsible for automated driving. This fact is a strong argument to introduce the same interface in simulation in order to enable seamless development, validation and testing within the development process as illustrated in Fig. 1.

Therefore, the sensor model proposed in this article also aims in using a generic *object-list*-like perception-interface in order to establish compatibility with various environment simulation frameworks on the one side, and the automated driving software on the other side. The main task of the sensor model will be to filter the perceived environmental information by either transmitting, cancelling or manipulating data within the *object-list*.

The paper is structured as follows. Section 2 summarizes the main requirements taken into account for the later definition of interfaces

**Stolz, Michael**, VIRTUAL VEHICLE Research Center, Inffeldgasse 21A, 8010 Graz, Austria (E-mail: [michael.stolz@v2c2.at](mailto:michael.stolz@v2c2.at)); **Nestlinger, Georg**, VIRTUAL VEHICLE Research Center, Inffeldgasse 21A, 8010 Graz, Austria (E-mail: [georg.nestlinger@v2c2.at](mailto:georg.nestlinger@v2c2.at))



**Fig. 1. Sensor models link environment and ego-vehicle simulation for developing and validating highly automated driving**

and generic sensor model. Section 3 gives a proposal for the interfaces and Sect. 4 introduces a generic sensor model, which may cover standard sensors used in automated vehicles such as radar, LiDAR, ultra-sonic and camera.

## 2. Requirements

Considering today's available sensors, one can make the following assumptions:

- (A1) Usually, one or more sensors are mounted to the vehicle to cover all interesting areas.
- (A2) Most sensors are "smart sensors", which have their own object detection, pre-processing of raw data and deliver an object-list at their output interface.
- (A3) Sensors have a specific field of view or detection area. This area may be approximated by a closed curve without intersections with itself.
- (A4) In simulations, ground truth data of all objects surrounding the vehicle under test is available. Sensor models can access this data directly from the environment simulation.

To meet the assumptions listed above, the proposed Generic Sensor Model (GSM) has to fulfill the following requirements:

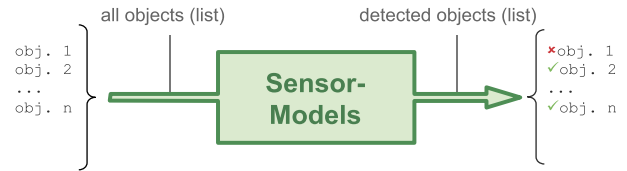
- (R1) The GSM should be modular and allow multiple sensors to be specified.
- (R2) The GSM (representing a smart sensor) has an object-list as output interface.
- (R3) The GSM provides a set of parameters to define a specific detection area.
- (R4) The GSM receives a ground truth object-list<sup>1</sup> as input.

## 3. Interfaces

Following the above requirements (R2) and (R4) and the common co-simulation approach of HAVs drawn in Fig. 1, *object-lists* are used as interfaces for sensor models. Figure 2 illustrates the working principle. All objects respectively their properties are input into the sensor model. The GSM modifies this object-list according to the object's visibility, i. e. objects are rated as detected or not detected.

In order to be able to attach multiple instances of the GSM representing different sensor setups to the same input signal, object properties have to be represented in the same coordinate system. We suggest to use the moving ego-vehicles coordinate system. Therefore a transformation may be needed from fixed *world-coordinate system* to the moving ego-vehicle reference frame using the ego-vehicles position, orientation and motion.

<sup>1</sup>An *object-list* is meant to be an aggregation of data sets, where each data set specifies the state of an object. The elements of these data sets are referred to as the properties (e. g. position, velocity, dimensions, ...) of the object.



**Fig. 2. Object-lists as interfaces for a generic sensor model for environment perception**

In the following, we propose a set of properties to be part of the object-list of the GSM's input and output. For the sake of simplicity we use basic data arrays instead of nested objects and class definitions. Therefore, an object-list in the scope of this work is defined as a matrix of doubles of size  $a \times b$ , where  $a$  is the number of properties and  $b$  is the number of objects. The choice of properties actually transmitted is up to the user. We use the following properties (relative to ego-vehicle reference frame):

- Object position in longitudinal direction ( $x$ ) in [m].
- Object position in lateral direction ( $y$ ) in [m].
- Object velocity over ground in longitudinal direction in [ $m s^{-1}$ ].
- Object velocity over ground in lateral direction in [ $m s^{-1}$ ].
- Object width in [m].
- To enable fixed sizes of input and output arrays, a status flag for each object representing one of the states {not detected, detected, newly detected} was used. The differentiation between "detected" and "newly detected" is important when fusing the outputs of multiple GSMs to initialize new tracks and filters [10].

## 4. Generic sensor models

Many tools used for simulating HAVs use either coarse geometric models (sector of a circle) or highly complex sensor models (ray-tracing based). Usually, simple geometric models are computationally rather cheap to evaluate, but do not offer enough flexibility to model detection ranges in detail. In contrast to this, very detailed sensor specific models typically require high computation power [4, 6]. Also, they are often used for sensor hardware development and testing of low level data-processing.

In the following, we propose a lean GSM, which is computationally cheap to execute but still offers big flexibility to apply it for modelling camera-, radar-, LiDAR- and ultra-sonic sensors. We address two main tasks of the sensor model within the simulation, namely *object detection* and *line of sight*, which are explained in the following.

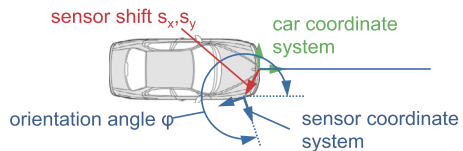
### 4.1 Sensor reference frame

In order to simplify later model definitions, a transformation of an object's position  $[d_x \ d_y]_{Car}^T$  in the ego-vehicle coordinate system to the sensor coordinate system is needed. This can be easily done within each GSM by a translation and a rotation

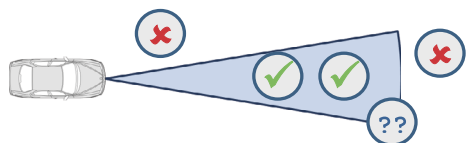
$$\begin{bmatrix} d_x \\ d_y \end{bmatrix}_{Sensor} = \begin{bmatrix} \cos \varphi & \sin \varphi \\ -\sin \varphi & \cos \varphi \end{bmatrix} \left( \begin{bmatrix} d_x \\ d_y \end{bmatrix}_{Car} - \begin{bmatrix} s_x \\ s_y \end{bmatrix}_{Car} \right) \quad (1)$$

using the sensor's mounting positions  $s_x$  and  $s_y$  and mounting orientation  $\varphi$  as shown in Fig. 3.

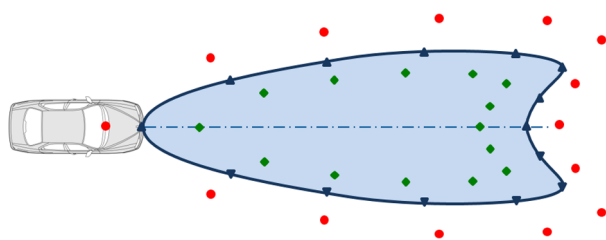
As a result, the sensor model parameters used for deciding, if an object is detected or not, become independent from the sensor's mounting position and orientation. Since there is no relative movement between the sensor and vehicle coordinate system, transformation of velocities is done in the same way.



**Fig. 3. Parameters of sensor mounting used to transform position, orientation and velocities from vehicle to sensor coordinate system**



**Fig. 4. Object detection: Sensor model decides if an object is in range or not. Partly overlapping objects need special handling**



**Fig. 5. Example of points  $x_i \in \mathbb{R}^2$  defining the detection area of a sensor: blue triangles are on the border ( $z_i = 1$ ), green rhombuses ( $z_i = 2$ ) are inside the sensor detection range, red dots ( $z_i = 0$ ) are outside (Color figure online)**

**4.2 Object detection**

Within the object detection, the sensor model *decides*, if an object is detected/not detected (see Fig. 4), assuming optimal conditions and no obstacles blocking the sensor’s line of sight.

The simplest approach to solve the detection task is to check, if the object is within the detection range. The result may be extended by statistic and/or phenomenological modelling parts. However, we focus on the geometric decision of detection only. The proposed

modelling approach is based on so called radial basis functions [3], which are commonly used for approximating and interpolating scattered data. The main idea is to provide a function  $z(x) : \mathbb{R}^2 \rightarrow \mathbb{R}$ , which classifies a specific point  $x \in \mathbb{R}^2$  as a potentially detected object by deciding if the point is within or outside the detection border:

$$z(x) = \begin{cases} < 1 & x \text{ is outside the sensor range,} \\ 1 & x \text{ is at the border of sensor detection,} \\ > 1 & x \text{ is inside the sensor range.} \end{cases} \quad (2)$$

In a first step the detection border of the sensor model is defined by introducing a set of  $n$  points  $x_i \in \mathbb{R}^2$  ( $i = 1, \dots, n$ ) as illustrated in blue in Fig. 5. In a second step this set of points is extended by a set of points inside the sensor detection range (green) and outside the sensor detection range (red). As described in the figure’s caption, values  $z_i \in \mathbb{R}$  are defined for each  $x_i$ .

Assuming all three sets having the same size of  $n$  elements, the model is finally evaluated using

$$z(x) = \sum_{i=1}^{3n} \lambda_i \phi(x, x_i) \quad (3)$$

$$\phi(x, x_i) = e^{-\frac{\|x - x_i\|_2^2}{\sigma^2}}, \quad (4)$$

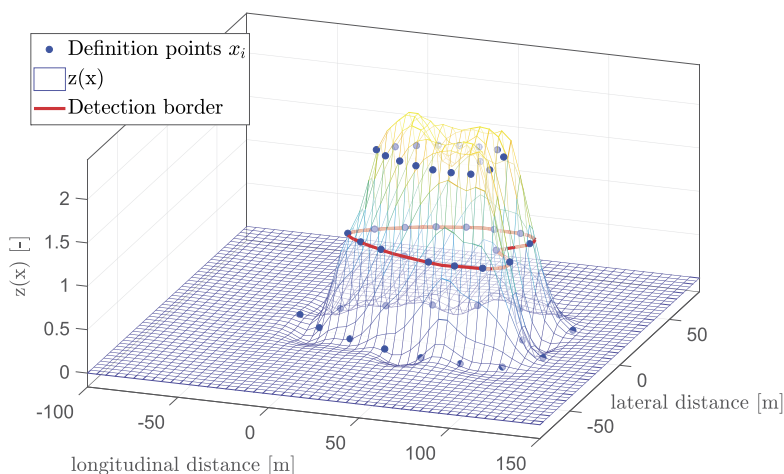
where  $\sigma$  is a scaling factor, and  $\lambda_i \in \mathbb{R}$  ( $i = 1, \dots, 3n$ ) are parameters derived once. The calculation of all  $\lambda_i$  is done in a single step in vector notation:

$$\lambda = (\Phi^T \Phi + \eta I)^{-1} \Phi^T z_d. \quad (5)$$

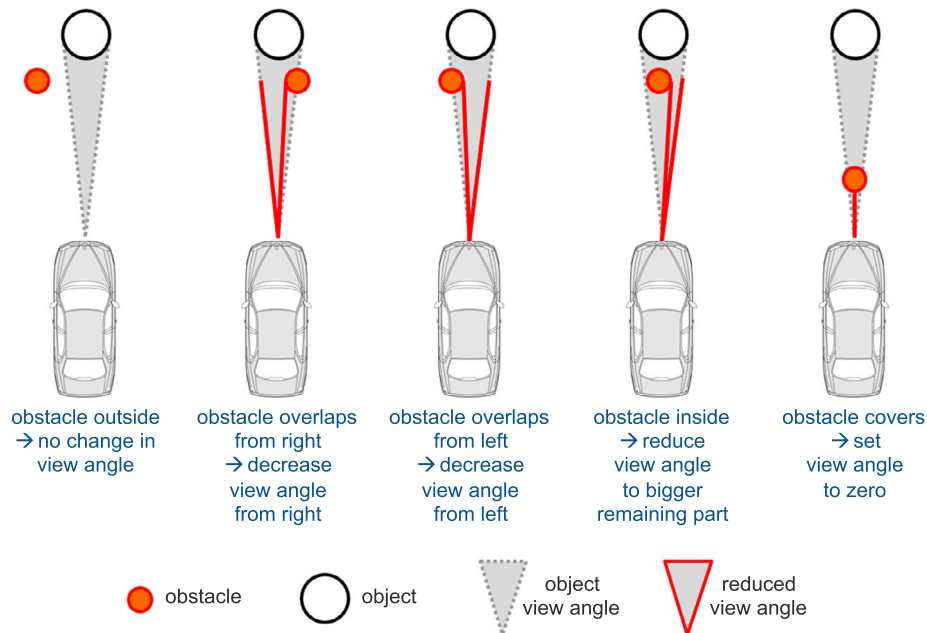
The matrix  $\Phi$  is of size  $3n \times 3n$  and its components of row  $i$  and column  $j$  are given by

$$\Phi_{i,j} = \phi(x_j, x_i). \quad (6)$$

Symbol  $\eta$  is a smoothing factor defining the trade-off between interpolation ( $\eta = 0$ ) and approximation ( $\eta > 0$ ). Vector  $z_d$  is of length  $3n$  and holds the desired values of the sensor model at the points  $x_i$  ( $i = 1, \dots, 3n$ ) defined in the first and second step above. Symbol  $I$  denotes the identity matrix of the same size as  $\Phi$ . Note that tuning  $\sigma$  is straight forward if the distances between neighboring points  $x_i$  are approximately equal (start with a value for  $\sigma$  equal to the distance of neighboring points). Figure 6 shows (3) for an exemplary



**Fig. 6. Sensor model evaluated in the surroundings of the vehicle. Values  $z > 1$  indicate the location within the sensor’s range of detection (red line). Blue dots mark the definition points  $x_i$  of the sensor model**



**Fig. 7.** Cases of possible configurations in determining, if objects are hidden or still visible. The angle covered by the object under test is reduced by obstacles in the line of sight

```

procedure SHADOWING(object-list)
  sort objects in descending distance
   $N =$  number of objects
  for  $n = 1$  to  $N - 1$  do
    for  $m = n + 1$  to  $N$  do
      if  $m$  is inside of  $n$ 
        or  $m$  totally hides  $n$ 
        or  $m$  partly hides  $n$  with its left side
        or  $m$  partly hides  $n$  with its right side then
          apply the reduction of visibility
        end if
      end for
    end for
    decide if object  $n$  is still visible
  end for
end procedure

```

**Fig. 8.** Simple algorithm for making a decision on the visibility of an object

GSM evaluated around the vehicle. Values greater 1 mark locations where objects get detected.

#### 4.3 Line of sight

To determine, if objects are hidden by other objects, the line of sight will be checked for all objects, which are in the sensor's field of view.

For evaluating the visibility, in the following each object is modeled as a circle with a defined radius (object width). Two lines of sight are drawn from the centre of the sensor's coordinate system tangentially to the object's circle as shown in Fig. 7. The so called *view angle* between these two lines is used in the algorithm shown in Fig. 8 to decide the visibility. Depending on the case, how obstacles cover this view angle, the view angle is reduced step by step by all obstacles which are between the object and the sensor. Finally, if the ratio of the uncovered angle and the reduced angle is still above a defined threshold (e.g. 20 %), the object is detected.

#### 4.4 Specifics of sensor principle

Sensor specific detection of defined object properties can be realized by either passing or blocking information from the object-list depending on the type of sensor (or actual environmental conditions). At this point, no full set of rules will be given, but the following examples shall give an impression how a specific implementation could look like:

- If the GSM simulates a radar sensor, braking lights status information will be blocked
- If the GSM simulates a camera sensor, braking lights status information will be transmitted.

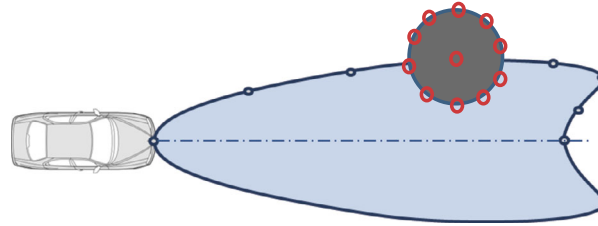
#### 4.5 Extensions

Up to now only 0D-objects have been treated in the object detection above. A possible extension to also handle the detection of objects with specific geometry, is to represent their geometry by a set of points and define e.g. a threshold by a number of points, that have to be inside the sensor detection area for positive detection (see Fig. 9).

#### 5. Summary and conclusion

This article summarized the role and requirements of generic sensor models for testing HAVs using simulation. The reader was introduced to the use of object-lists as state of the art interfaces. A detailed approach for fast and flexible geometric sensor models has been given together with a straightforward algorithm for checking line of sight. Possible extensions to more complex object geometries have been discussed.

The proposed GSM offers an effective and efficient way for developing and testing control algorithms of HAVs in simulation. The ability of defining a nearly arbitrary sensor detection geometry is a big advantage of the proposed approach compared to standard sector-based geometries. The computationally cheap evaluation enables the use in accelerated tests. Future focus will be on standardizing the interfaces, i.e. the minimal content of object-lists.



**Fig. 9.** Model extension for detecting not only 0D-objects but also 2D-objects is straight forward, but increases computational effort

### Acknowledgements

Open access funding provided by Graz University of Technology. This work was accomplished at the VIRTUAL VEHICLE Research Center in Graz, Austria. This work has been partially supported in the framework of the Austrian COMET-K2 program. This work has been funded by FFG under the project Traffic Assistant Simulation and Traffic Environment—TASTE, project number 849897. The authors further want to thank and acknowledge the whole TASTE project team for the assistance, fruitful discussions and suggestions during the course of this work.

**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

### References

1. AVL LIST GmbH AVL CRUISE™—vehicle driveline simulation.
2. AVL LIST GmbH AVL VSM 4™—vehicle dynamics simulation.
3. Buhmann, M. (2003): Radial basis functions: theory and implementations. Cambridge monographs on applied and computational mathematics. Cambridge: Cambridge University Press.
4. Hanke, T., Hirsenkorn, N., Dehlink, B., Rauch, A., Rasshofer, R., Biebl, E. (2015): Generic architecture for simulation of adas sensors. In 2015 16th international radar symposium, IRS (pp. 125–130). <https://doi.org/10.1109/IRS.2015.7226306>.
5. Hanke, T., Hirsenkorn, N., van Driesten, C., Garcia-Ramos, P., Schiementz, M., Schneider, S. K. (2017): Open simulation interface—a generic interface for the environment perception of automated driving functions in virtual scenarios.
6. Hirsenkorn, N., Hanke, T., Rauch, A., Dehlink, B., Rasshofer, R., Biebl, E. (2016): Virtual sensor models for real-time applications. *Adv. Radio Sci.*, 14, 31–37. <https://doi.org/10.5194/ars-14-31-2016>. URL: <https://www.adv-radio-sci.net/14/31/2016/>.
7. IPG Automotive GmbH: CarMaker
8. Krajzewicz, D., Erdmann, J., Behrisch, M., Bieker, L. (2012): Recent development and applications of SUMO—simulation of urban MOBility. *Int. J. Adv. Syst. Measur.*, 5(3&4), 128–138.
9. PTV Planung Transport Verkehr AG: Vissim.
10. Stüker, D. (2004): Heterogene Sensordatenfusion zur robusten Objektverfolgung im automobilen Straßenverkehr. Phd, Carl von Ossietzky-Universität Oldenburg. URL: <http://d-nb.info/972494464/34>.
11. VIRES Simulationstechnologie GmbH: VTD—virtual test drive.
12. Watzenig, D., Horn, M. (Eds.) (2016): Automated driving: safer and more efficient future driving. Berlin: Springer. <https://doi.org/10.1007/978-3-319-31895-0>.
13. Winner, H., Hakuli, S., Lotz, F., Singer, C. (Eds.) (2016): Handbook of driver assistance systems: basic information, components and systems for active safety and comfort. Switzerland: Springer. <https://doi.org/10.1007/978-3-319-12352-3>.

### Authors



#### Michael Stolz

After finishing his studies in mechanical engineering at Graz University of Technology, Graz, Austria, in 2002, Michael Stolz worked for six years in automotive industry at AVL-List GmbH as development engineer for diesel control algorithms. In 2009 he changed to the virtual vehicle research center elaborating his Ph.D. thesis within a founded cooperation project on the topic of diesel control, which

he finished at the Institute of Automation and Control at Graz University of Technology in 2013. Since 2014 he has been responsible for the Control Systems group at the virtual vehicle research center.

Main working areas are control-oriented real time modeling; automotive control and control architectures; automated driving.



#### Georg Nestlinger

Studied electrical engineering at Graz University of Technology, Graz, Austria and received his diploma in 2013. After his studies he was with AutomationX, an international technology company focusing on integrated automation solutions. He then joined the virtual vehicle research center in Graz as a junior researcher. His main research interests are in the field of automated driving and control engineering.