


A Symbolic Design Method for ETCS Hybrid Level 3 at Different Degrees of Accuracy

Stefan Engels¹ ✉ 

Chair for Design Automation, Technical University of Munich, Germany

Tom Peham ✉ 

Chair for Design Automation, Technical University of Munich, Germany

Robert Wille ✉  

Chair for Design Automation, Technical University of Munich, Germany

Software Competence Center Hagenberg GmbH (SCCH), Austria

Abstract

The *European Train Control System (Hybrid) Level 3* (ETCS Hybrid Level 3) allows for introducing *Virtual Subsections* (VSS) into existing railway infrastructures. These VSS work similarly to blocks in conventional block signaling but do not require installation or maintenance of trackside train detection. This added flexibility can be used to adapt a given railway network’s (virtual) layout to the changing demands of new schedules. Automated methods are needed to properly use this flexibility and design such layouts on demand and avoid time-intensive manual labor. Recently, approaches inspired by design automation of electronic hardware have been proposed to address this need. But those methods—which are particularly well suited for inherently discrete problems in electronic design automation—have struggled with modeling continuous properties like train positions, time, and acceleration. This work proposes a *Mixed Integer Linear Programming* (MILP) formulation that, for the first time, can accurately model design problems for ETCS Hybrid Level 3 by including essential, continuous constraints, e.g., for train dynamics or braking curves. The formulation is designed to be flexible and extendable, allowing the user to include/exclude certain constraints or simplify the model as needed. By this, the user can decide whether he/she wants to quickly generate a less accurate solution or a more accurate one at the expense of higher runtimes—basically allowing him/her to trade-off accuracy and efficiency. A case study showcases the potential of the proposed approach and sketches examples to analyze which trade-offs are worthwhile and which simplifications can be safely made. The resulting tool and the benchmarks considered in this work are publicly available at <https://github.com/cda-tum/mtct> (as part of the *Munich Train Control Toolkit*, MTCT).

2012 ACM Subject Classification Applied computing → Transportation

Keywords and phrases ETCS, MILP, design automation, block signaling, virtual subsection

Supplementary Material *Software (Source Code)*: <https://github.com/cda-tum/mtct>

1 Introduction

Although railway transportation has a long history, it also plays a vital role in the future of sustainable transportation. Unlike cars, trains cannot be operated on sight, and signaling systems are essential to prevent collisions. For this, many national train control systems have been implemented—about 40 in Europe alone [13].

Due to the resulting compatibility issues, especially in times of increasing cross-border traffic, it was decided to harmonize these safety systems across Europe, namely in the *European Train Control System* (ETCS) [30]. Similar standardized systems exist in China (*Chinese Train Control System*, CTCS), North America (*Positive Train Control*, PTC), and even for metro lines (*Communication Based Train Control*, CBTC) [25, 29].

¹ Corresponding author

44 In addition to harmonization, these systems also aim at increasing throughput and, by
45 this, increase demand for the entire train infrastructure. In fact, if building new tracks is not
46 feasible, the increasing demand for rail transportation has to be tackled another way. To this
47 end, new levels of train control systems have constantly been defined to allow for shorter train
48 following times while maintaining the high safety requirements imposed [27]. *ETCS Hybrid*
49 *Level 3* (ETCS HL3), for instance, defines *Virtual Subsections* (VSS) that introduce new
50 blocks into an existing layout without requiring new hardware. In addition to positioning
51 being determined via *Trackside Train Detection* (TTD) systems such as axle counters, the
52 occupation of VSS is communicated live via a radio control center. This, in turn, allows
53 for a much more fine-grained design of the train control system since the overhead does not
54 limit the number of blocks for maintenance and installation of TTDs, and the block layout
55 can be changed on demand as it is only virtual. In principle, such virtual layouts also allow
56 for shorter headways and can, therefore, be used to increase the throughput of an existing
57 network.

58 This potential gives rise to several new design tasks for ETCS HL3 systems, namely *veri-*
59 *fication* of HL3 layouts, *placement* of VSS, and *optimization* of train schedules using VSS [11].
60 Previous methods for designing ETCS L2 layouts did not have to consider dynamic block place-
61 ment and are, therefore, aimed at more general performance indicators [14, 5, 18, 9, 23, 31].
62 Similarly, while train routing [15] and allocation [6, 3, 21, 4, 22] have been considered, these
63 approaches are tailored for fixed layouts. There is also work on routing under ETCS Level 3,
64 which uses so-called *moving blocks* and does not require *any* VSS.

65 Design tasks within ETCS HL3 have already been tackled using symbolic reasoning [32]
66 and guided state-space search [26]. While these approaches seem promising for these tasks,
67 they all make simplifying assumptions and do not model train movement—in particular
68 acceleration and braking curves—accurately. This is partly due to the fundamental limitations
69 of the methods used, as they are ill-fitted to model continuous properties directly.

70 *Mixed Integer Linear Programming* (MILP) is an alternative symbolic method that allows
71 for modeling discrete values as well as continuous variables and is, therefore, a promising
72 approach for solving design problems for ETCS HL3 accurately. MILP has already been used
73 previously for routing within ETCS Level 3 with full moving blocks [28, 19], but no MILP
74 approach exists that can automatically design ETCS HL3 layouts. This work introduces
75 a comprehensive framework for solving ETCS HL3 design tasks using a symbolic MILP
76 formulation that enables a designer of ETCS HL3 networks to model problems with varying
77 degrees of accuracy. For the first time, this framework allows for accurate modeling of
78 continuous properties for ETCS HL3 design tasks. Furthermore, since the expressiveness of
79 MILP also subsumes previous formulations, the proposed MILP formulation can be simplified
80 or extended with further constraints to model ETCS HL3 design tasks with varying degrees
81 of accuracy.

82 It is to be expected that more detailed models are harder to solve and, thus, lead to longer
83 solving times. The flexibility of the proposed framework allows for evaluating how model
84 accuracy influences runtimes for solving instances of ETCS HL3 design tasks. Therefore, the
85 impact of using more or less detailed models is evaluated on a range of benchmarks, including
86 real-world examples, e.g., designing an ETCS HL3 layout for the S-Bahn Stammstrecke in
87 Munich. The framework and the benchmarks are publicly available within the *Munich Train*
88 *Control Toolkit* (MTCT) at <https://github.com/cda-tum/mtct>.

89 The remainder of this paper is structured as follows: Sec. 2 reviews the relevant background
90 on block-signaling within ETCS and the design problem considered in this work. Sec. 3 then
91 proposes the MILP formulation, starting with a minimal required encoding and successively
92 introducing constraints that allow for more accurate models. The evaluation of the trade-off
93 between runtime efficiency and model accuracy is discussed in Sec. 4. Finally Sec. 5 concludes
94 this paper.

2 Block Signaling in ETCS HL3

ETCS HL3 builds upon principles of classic block signaling. This section provides the necessary background on block signaling within ETCS and what constraints exist on VSS placement.

2.1 Background

ETCS Level 1 (ETCS L1) separates a railway network in blocks. *Trackside Train Detection* (TTD), e.g., *Axle Counters* (AC), at the boundaries is used to determine if a train is present within a certain block. Hence they are also known as TTD sections. Conventional signals are used to show if the upcoming block is occupied or not. At distinct points, the signal state is transmitted to the train through *Eurobalises* (EB). A train always has to be able to come to a complete stop before the point to which it has received moving authority to, which is ensured by braking curves [12].

In *ETCS Level 2* (ETCS L2), trains communicate with the control system via the wireless system GSM-R. Balises are no longer used to transmit variable information, but fixed position data is employed instead. Still, TTDs are used to detect the status of blocks. Move authority is continuously transmitted to the trains.

With the introduction of *ETCS Level 3* (ETCS L3) the main principles change for the first time since the 19th century. The train itself reports the exact position and its integrity to the control system. TTDs are no longer needed to safely declare track segments as free. In theory, this allows movement in a so-called *moving block*, i.e., trains are cleared to drive all the way to the previous train (minus some safety buffer) without the necessity of any fixed blocks or correspondingly needed TTD systems.

Since the main principle of block signaling is not part of L3, it poses difficulties when implemented in practice [2]. Because of this, *ETCS Hybrid Level 3* (ETCS HL3) has been specified in [10] to overcome this issue, yet providing the advantages of Level 3 systems. Again the trains transmit their location and integrity to the control system, and TTDs are not necessary to clear sections. Hence, existing TTD sections can be divided into *Virtual Subsections* (VSS) without adding additional hardware. The principles of L2 remain the same, with the only exception that these new VSS are used instead of the old sections, which, again, allows shorter train following times than on low-level systems.

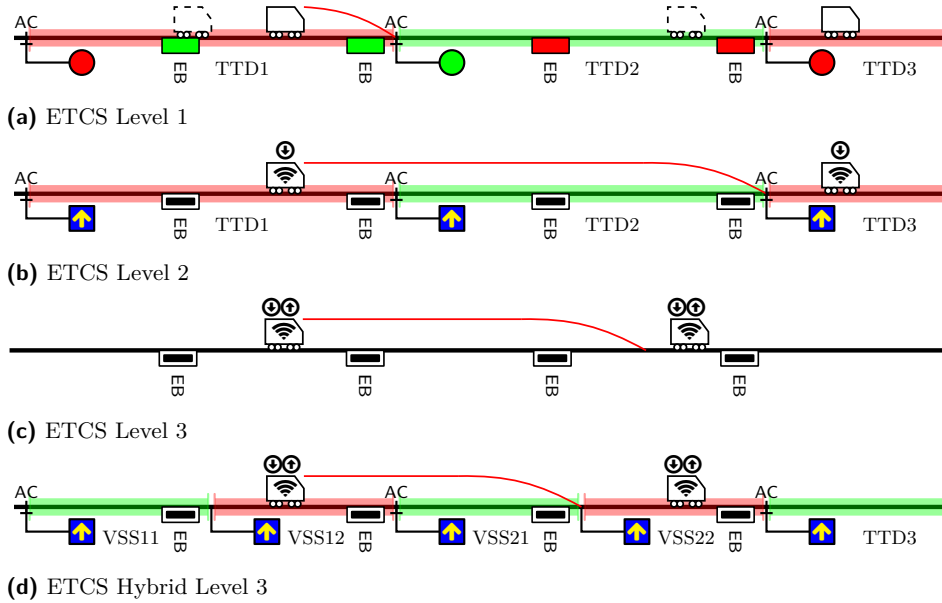
► **Example 1.** Consider two trains following one another on a straight section of track as shown in Fig. 1 for different ETCS Levels.

Fig. 1a shows how block-signaling works in ETCS Level 1. While the train on the right has already cleared TTD2, the following train has not yet received permission to enter TTD2 because the block was still occupied when the train last passed a balise. As soon as it hits the next balise (this time transmitting a green signal), the authority is updated so that the following train can enter TTD2.

This problem does not exist in ETCS Level 2 as the following train receives permission to enter TTD2 immediately after the train on the right has left that block in Fig. 1b.

In Fig. 1c, the flexibility of ETCS Level 3 (using moving block) is shown. The train on the left can follow the leading train as close as possible, only needing to keep the respective braking distance.

Fig. 1d shows a compromise between Fig. 1b and Fig. 1c by separating TTD2 into two virtual subsections. This allows the train on the left to follow more closely without requiring the installation of additional hardware.



■ **Figure 1** Schematic drawings of various ETCS levels

140 2.2 Placing Virtual Subsections under ETCS HL3

141 In this work, we focus on the promising ETCS HL3, which allows for separating TTD sections
 142 into virtual subsections (VSS). Some TTD sections might not be separable into VSS, e.g.,
 143 because of turnouts (where close section borders would not comply with flank protection) or
 144 constraints imposed by railway crossings or section breaks in overhead lines [17]; on others
 145 only a minimal VSS length is specified.

146 Since VSS do not require new trackside hardware, they can, in theory, be changed without
 147 changing the trackside hardware on a virtual level. Hence, adapting the block layout for a
 148 new schedule might, for the first time, be reasonable. Because of this, it is of great interest
 149 to consider precise timetables and block layouts jointly in the planning process.

150 Various design tasks arise within the abovementioned context (see also [32, 26, 11]). Since
 151 adding VSS to preexisting railway networks can increase their capacity, it is of interest to
 152 efficiently determine where to place them, also with respect to a new timetable, which might
 153 not be realizable on a given TTD layout. One wants to find a VSS layout under which
 154 the previously infeasible timetable can be accomplished. Or, one might want to tweak the
 155 timetable to reduce the travel time or headway to a minimum using a predefined number of
 156 VSS. Finally, one can also consider a mixed mode of passenger and freight services. In that
 157 case, a predefined schedule should be fulfilled while maximizing freight train throughput.

158 Exemplary, in this case study, we focus on generating layouts to realize predefined
 159 schedules. We are given (part) of a railway network under consideration together with a
 160 (macroscopic) timetable for various trains. This includes times when trains enter and leave
 161 the network and scheduled stops in between. The number of VSS sections that can be
 162 implemented in operation is limited by the efficiency of the used components. Hence, it
 163 is desirable to keep the number of VSS low. If the control system in operation can safely
 164 manage more sections, the remaining ones might be used to improve robustness. Overall this
 165 leads to the design task considered in this work: Given a railway network with TTD sections,
 166 a list of trains, and their respective (macroscopic) timetables, separate the TTD sections
 167 into a minimal number of VSS to make the timetable feasible and determine a respective
 168 (microscopic) routing/refinement.

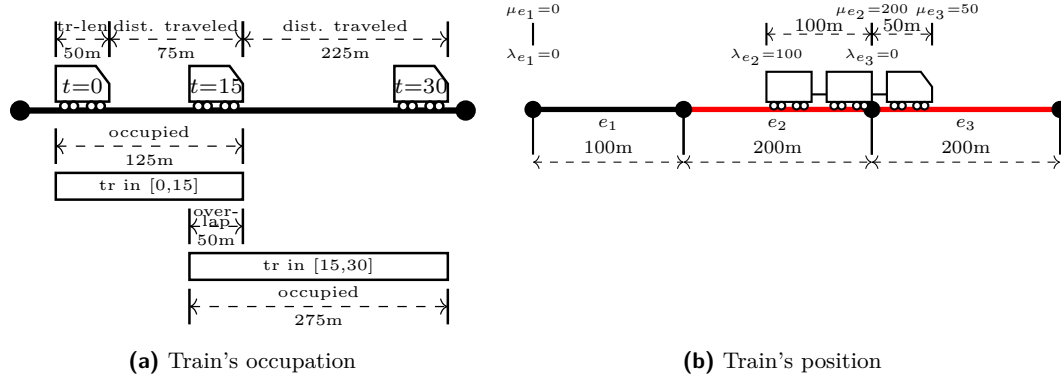


Figure 2 Modeling of continuous positions depending on routing choice

3 Symbolic Formulation

In the literature, various models have been introduced which, in combination with corresponding reasoning engines or solvers, can be used to solve the design task described in Sec. 2.2 [32, 26]. In this work, we propose a method that allows for a trade-off between the accuracy and efficiency of such formulations. We start with a base formulation that contains only the most relevant details on an equivalent level to previous work (Sec. 3.1). Afterward, we describe how more realistic, continuous details like train dynamics (Sec. 3.2) and braking curves enforced by the train control system (Sec. 3.3) can be added to the base formulation. Finally, we consider how fixing the train routes a priori simplifies the described model (Sec. 3.4).

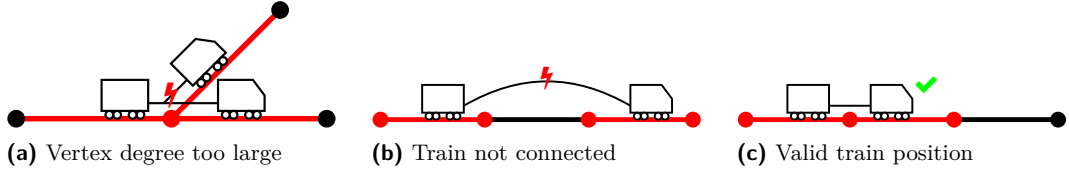
We keep the corresponding descriptions at a high level yet precise enough to allow for a discussion in the following sections. In particular, we omit constraints irrelevant to understanding the central ideas or whose logical form is easier to grasp the relevant concepts. They can easily be transformed into linear constraints using standard techniques (e.g., big-M). Readers interested in a more detailed treatment are referred to our open-source implementation available at <https://github.com/cda-tum/rail>, which includes some minor (yet efficient) additions to strengthen the relaxation.

3.1 Base Model

To state a MILP model, we first discretize the time horizon into intervals of a predefined length, say Δt . Train positions are then modeled over intervals instead of single time points. Assume that v_t and $v_{t+\Delta t}$ are the velocities of a train at the interval boundaries of $[t, t + \Delta t]$ and that the speed of the train within the interval can be determined by linearly interpolating between the initial and final velocity. Then, the traveled distance can be easily approximated as $\frac{v_t + v_{t+\Delta t}}{2} \cdot \Delta t$. In particular, during a given time interval, a train occupies not only the track corresponding to its length but also the track section it travels over. The intersection of occupied track sections at two adjacent intervals $[t - \Delta t, t]$ and $[t, t + \Delta t]$ leads to the exact position at time point t .

► **Example 2.** Consider Fig. 2a with a 50m-long train moving forward in time steps of 15 seconds. At $t = 0$ s the train stopped and is now accelerating to 10m/s at $t = 15$ s and 20m/s at $t = 30$ s. By assuming that the velocity can be linearly interpolated (e.g., $v_{7.5s} = 5$ m/s), the train moves 75m within $[0s, 15s]$ and 225m within $[15s, 30s]$. Hence, it occupies a total of 50m + 75m = 125m in the first interval and 50m + 225m = 275m in the latter. The overlap of the two occupations has a length of 50m, i.e., the same length as the train.

In the following paragraphs, we describe how this basic idea is implemented in the MILP model.



■ **Figure 3** Ensuring trains position is valid

204 **Variables describing train positions:** To model the abovementioned, we add the following
 205 variables to the model:

206 ■ $v_t^{tr} \in [0, v_{max}^{tr}]$: is the current speed of train tr (with maximal speed v_{max}^{tr}) at time t .

207 ■ $x_{t,e}^{tr} \in \{0, 1\}$: indicates if train tr occupies edge e anytime within $[t, t + \Delta t]$.

208 A train does usually not occupy an entire edge but might only be present on parts of it.
 209 Thus, train positions cannot be modeled precisely using only binary (discrete) variables ². In
 210 a MILP setting, continuous variables can also be added. By doing so, we model the exact
 211 train position on an edge using variables for both the front and rear of the train:

212 ■ $\mu_{t,e}^{tr} \in [0, \text{len}(e)]$: front of tr on e measured from edge's start in interval $[t, t + \Delta t]$.

213 ■ $\lambda_{t,e}^{tr} \in [0, \text{len}(e)]$: rear of tr on e measured from edge's start in interval $[t, t + \Delta t]$.

214 The binary variables $x_{t,e}^{tr}$ indicate that a train is present on edge e . These can be inferred
 215 from $\mu_{t,e}^{tr}$ and $\lambda_{t,e}^{tr}$. If they are both equal to 0, the train is not present on the respective
 216 edge; otherwise, it is.

217 ► **Example 3.** Consider the simple network shown in Fig. 2b with a 150m-long train located
 218 on edges e_2 and e_3 . Both these edges are 200m long, but the train only occupies 100m and
 219 50m, respectively. In this case, $\lambda_{e_2} = 100\text{m}$ and $\mu_{e_2} = 200\text{m}$ denote that the train occupies
 220 the second half of e_2 . Similarly, $\lambda_{e_3} = 0\text{m}$ and $\mu_{e_3} = 50\text{m}$. Because the train is not present
 221 anywhere on e_1 , we have $\lambda_{e_1} = \mu_{e_1} = 0\text{m}$. This implies that $x_{e_2} = x_{e_3} = 1$, i.e. the train
 222 occupies edge e_2 and e_3 . On the other hand, $x_{e_1} = 0$, i.e., the train does not occupy e_1 .

223 **Occupation and overlap:** The length of the track section a train occupies during one
 224 timestep can be obtained by summing over the $(\mu_e - \lambda_e)$ -differences over every edge e the
 225 train occupies. The overlap length is obtained by taking the difference of these lengths for
 226 two adjacent time steps (given that the train is present on the edge). Symbolically, these
 227 constraints are encoded in the MILP formulation as follows:

$$228 \quad \sum_{e \in E} (\mu_{t,e}^{tr} - \lambda_{t,e}^{tr}) = \text{len}(tr) + \frac{v_t^{tr} + v_{t+\Delta t}^{tr}}{2} \cdot \Delta t \quad \forall t, tr \quad (1)$$

$$229 \quad \sum_{e \in E} x_{t+\Delta t,e}^{tr} (\mu_{t,e}^{tr} - \lambda_{t+\Delta t,e}^{tr}) = \text{len}(tr) \quad \forall t, tr. \quad (2)$$

230 **Train integrity:** The formulation above does not guarantee valid train positions without
 231 further constraints. For example, the situations depicted in Fig. 3a and Fig. 3b would be
 232 technically correct, as the length constraints are not violated.

233 Let $G = (V, E)$ be the graph representing a railway network, and for any vertex $v \in V$,
 234 let $\delta(v)$ denote the set of incident edges of v . Then the situation depicted in Fig. 3a can be
 235 avoided by imposing

$$236 \quad \sum_{e \in \delta(v)} x_{t,e}^{tr} \leq 2 \quad \forall v \in V, \quad (3)$$

² This is already a departure point from previous symbolic approaches that encoded positions as binary variables indicating whether a train occupies an edge.

237 i.e., train tr can occupy at most two adjacent edges to any vertex during any given time t .

238 The situation in Fig. 3b can be avoided by utilizing the following observation: for any
 239 cycle-free subgraph $G' = (V', E')$ of a railway network, being connected is equivalent to
 240 $|E'| = |V'| - 1$. Since the edges a train occupies during one timestep form a subgraph of G ,
 241 we can encode this constraint on the cardinalities of E' and V' as follows:

$$242 \quad \sum_{e \in E} x_{t,e}^{tr} = \sum_{v \in V} \left(\bigvee_{e \in \delta(v)} x_{t,e}^{tr} \right) - 1 \quad \forall t, tr, \quad (4)$$

243 for all trains tr , times t . Assuming all cycles within the railway network are sufficiently large
 244 (i.e., longer than the train's length plus maximal possible braking distance), the subgraph
 245 induced by the edges the train occupies in one timestep is cycle-free by design.

246 Note that more constraints are required to ensure realistic modeling of train movements
 247 (e.g., trains can only move in the direction of the engine). For brevity's sake, we omit these
 248 here.

249 **Speed limit:** While every train's maximal speed is directly included in the variable bounds,
 250 there might be a more restrictive speed limit on some railway tracks. Hence the following
 251 logical constraints need to be imposed:

$$252 \quad x_{t,e}^{tr} = 1 \Rightarrow v_t^{tr} \leq v_{max}^e \text{ and } v_{t+\Delta t}^{tr} \leq v_{max}^e \quad \forall t, e, tr. \quad (5)$$

253 **Timetable:** Trains are affiliated with a train schedule. In our setting, a schedule essentially
 254 defines if a train needs to stop at stations (i.e., a subset of edges of the railway network) at
 255 certain times. This includes the possibility to include multiple parallel tracks (corresponding
 256 to different platforms). Assume that tr has to stop in station $S_i^{tr} \subset E$ during the time
 257 interval $[\underline{t}_i^{tr}, \bar{t}_i^{tr}]$. Obviously, this means that the train must have a speed of 0m/s during that
 258 time interval which is encoded by the constraint

$$259 \quad v_t^{tr} = 0 \quad \forall t \in [\underline{t}_i^{tr}, \bar{t}_i^{tr}]. \quad (6)$$

260 Moreover, the train must be fully positioned within the station and cannot occupy any track
 261 outside the station, hence, we have

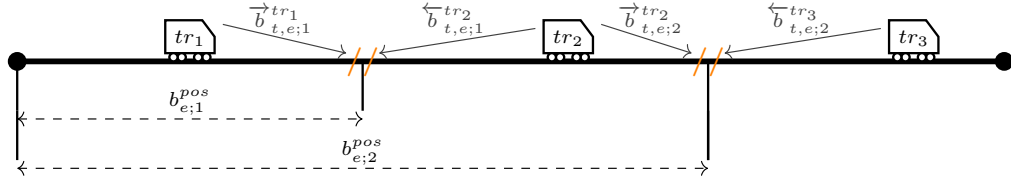
$$262 \quad x_{t,e}^{tr} = 0 \quad \forall t \in [\underline{t}_i^{tr}, \bar{t}_i^{tr}], e \in E - S_i^{tr} \quad \text{and} \quad \sum_{e \in S_i^{tr}} x_{t,e}^{tr} \geq 1 \quad \forall t \in [\underline{t}_i^{tr}, \bar{t}_i^{tr}]. \quad (7)$$

263 **VSS condition:** Finally, we model constraints imposed by an ETCS HL3 control system.
 264 In previous work, VSS boundaries were solely modeled by binary variables on vertices. In
 265 this paper, we model VSS boundaries as continuous variables instead. Assume that we allow
 266 I_e VSS boundaries to be set on an edge e , introduce variables $b_{e;i}^{pos} \in [0, \text{len}(e)]$ for $0 \leq i < I_e$
 267 denoting the positions of the respective VSS boundaries or 0 if they are not used. Hence, we
 268 can already formulate the objective of generating VSS layouts (using some small $\varepsilon > 0$, e.g.,
 269 the minimal VSS block length) on a logic level as

$$270 \quad \min \sum_{e \in E} \sum_{i \in I_e} [b_{e;i}^{pos} \geq \varepsilon] \quad ([\cdot]) \text{ denotes the Iverson bracket}. \quad (8)$$

271 Now, assume that n trains are present on one edge e at time t . Then they have to be
 272 separated by $n - 1$ VSS boundaries. Put differently, there must be $n - 1$ VSS boundaries
 273 that separate some trains' front from some other trains' rear. Let this be indicated by binary
 274 variables $\vec{b}_{t,e;i}^{tr}$ and $\overleftarrow{b}_{t,e;i}^{tr}$ respectively, then

$$275 \quad \vec{b}_{t,e;i}^{tr} = 1 \Rightarrow \mu_{t,e}^t \leq b_{e;i}^{pos} \quad \text{and} \quad \overleftarrow{b}_{t,e;i}^{tr} = 1 \Rightarrow \lambda_{t,e}^t \geq b_{e;i}^{pos} \quad \forall t, tr, e, i. \quad (9)$$



■ **Figure 4** Including VSS constraints in the model

276 Finally, it has to be ensured that the correct number of $\vec{b}_{t,e;i}^{tr}$ and $\overleftarrow{b}_{t,e;i}^{tr}$ is 1, so that $n - 1$
 277 VSS boundaries are chosen.

278 ► **Example 4.** Consider Fig. 4 with three trains on an edge. The trains are separated by
 279 two VSS boundaries, whose positions on the edge are given by the continuous variables $b_{e;1}^{pos}$
 280 and $b_{e;2}^{pos}$. The first VSS boundary separates tr_1 's front from tr_2 's rear ($\vec{b}_{t,e;1}^{tr_1} = \overleftarrow{b}_{t,e;1}^{tr_2} = 1$).
 281 Similarly, the second VSS boundary separates tr_2 from tr_3 ($\vec{b}_{t,e;2}^{tr_2} = \overleftarrow{b}_{t,e;2}^{tr_3} = 1$). All other
 282 binary indicators are 0 in this case.

283 The formulation described here allows for modeling the problem of VSS layout generation
 284 in comparable accuracy as previous work [32, 26]. However, in the following sections, we
 285 show how MILP can be used for modeling additional aspects that are infeasible or hard to
 286 encode in previous formulations due to their discrete nature.

287 3.2 Train Dynamics

288 In the base MILP model, train movements are not constrained by realistic dynamics like
 289 acceleration and deceleration. For a more realistic model, we also need to encode these
 290 properties. In fact, given maximal accelerations a_{tr} and decelerations d_{tr} , these dynamics
 291 can be added to the base MILP model with few constraints:

$$292 \quad v_{t+\Delta t}^{tr} \leq v_t^{tr} + \Delta t \cdot a_{tr} \quad \forall t, tr \quad \text{and} \quad v_{t+\Delta t}^{tr} \geq v_t^{tr} - \Delta t \cdot d_{tr} \quad \forall t, tr. \quad (10)$$

293 3.3 Braking Curves

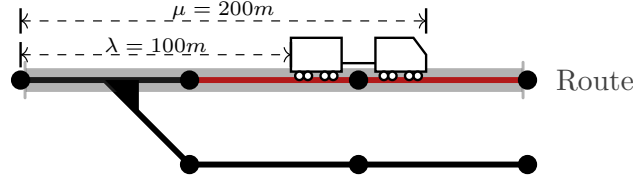
294 As described in Sec. 2, train control systems (such as ETCS) ensure that the complete track
 295 section a train needs to come to a full stop is not occupied by any other train using braking
 296 curves. The base model has no restrictions on safety distances between trains.

297 In time interval $[t, t + \Delta t]$, the final velocity is given by $v_{t+\Delta t}^{tr}$ and the braking distance
 298 of tr can be approximated using

$$299 \quad brakelen_t^{tr} = \frac{1}{2 \cdot d_{tr}} \cdot (v_{t+\Delta t}^{tr})^2 \quad \forall t, tr. \quad (11)$$

300 This is not linear (in the variable $v_{t+\Delta t}^{tr}$ to be precise), which poses problems with the
 301 inclusion in MILPs. Since it is an equality constraint, the resulting feasible region is not
 302 even convex. However, some solvers can even solve these constraints within a mixed integer
 303 program to optimality by using spatial branching [1, 24]. The approximation is then included
 304 locally in the respective branched subproblems. Hence, it is possible to model the braking
 305 distance directly.

306 Alternatively, one can add Eq. (11) as a piecewise linear approximation globally to
 307 the problem formulation itself. Under the hood, this will add binary variables and linear
 308 constraints. In particular, the integer model remains linear and can be solved with any
 309 MILP-solver.



■ **Figure 5** Modeling continuous train position on fixed routes

310 In either case, we add the braking distance to the length of the track section a train
 311 occupies by slightly adapting Eq. (1) and (2):

$$312 \quad \sum_{e \in E} (\mu_{t,e}^{tr} - \lambda_{t,e}^{tr}) = \text{len}(tr) + \frac{v_t^{tr} + v_{t+\Delta t}^{tr}}{2} \cdot \Delta t + \text{brakelen}_{t+\Delta t}^{tr} \quad \forall t, tr. \quad (12)$$

$$313 \quad \sum_{e \in E} x_{t+\Delta t,e}^{tr} (\mu_{t,e}^{tr} - \lambda_{t+\Delta t,e}^{tr}) = \text{len}(tr) + \text{brakelen}_t^{tr} \quad \forall t, tr.. \quad (13)$$

314 In some sense, we let the train length dynamically change throughout time depending
 315 on the speed. Since the train lengths attribute to the occupied track sections, no further
 316 constraints need to be added, as the base model already restricts these.

317 3.4 Fixed Routes

318 When designing a train schedule, a train's route (i.e., the exact tracks it uses) is sometimes
 319 already known a priori. If not, it might be possible to fix routes separately before placing
 320 VSS sections. In this case, the model significantly simplifies. If a train's route is known
 321 a priori, it is not necessary to model the exact location on every edge but can rather be
 322 modeled by single integer variables:

323 ■ $\mu_t^{tr} \in [0, \text{len}(\text{route})]$: front of the train in interval $[t, t + \Delta t]$

324 ■ $\lambda_t^{tr} \in [0, \text{len}(\text{route})]$: rear of the train in interval $[t, t + \Delta t]$

325 Since the position of the edges within a route is known, the corresponding indicator variables
 326 follow quickly. Variables corresponding to the exact position on every edge can even be
 327 removed entirely.

328 Additionally, Eq. (1) and (2) simplify to

$$329 \quad \mu_t^{tr} - \lambda_t^{tr} = \text{len}(tr) + \frac{v_t^{tr} + v_{t+\Delta t}^{tr}}{2} \cdot \Delta t \quad \forall t, tr \quad (14)$$

$$330 \quad \mu_t^{tr} - \lambda_{t+\Delta t}^{tr} = \text{len}(tr) \quad \forall t, tr. \quad (15)$$

331 Braking curves can be included analog to Eq. (12) and (13).

332 ► **Example 5.** Consider the network in Fig. 5 with a 100m-long train. While the train
 333 could potentially choose different routes, in theory, it is fixed to take the top track. Hence,
 334 $\lambda = 100m$ and $\mu = 200m$ uniquely define the train's position measured from the route's
 335 starting point.

336 4 Case Study

337 The symbolic formulation in its different modeling details presented in the previous section
 338 has been implemented in C++ and made publicly available as open-source implementation
 339 at <https://github.com/cda-tum/rail>. By that, a tool got available which allows the user
 340 to include/exclude certain constraints and, by that, decide whether he/she wants to quickly

341 generate a less accurate solution or a more accurate solution at the expense of higher runtimes.
 342 This section now summarizes the results of a case study showcasing the potential of such an
 343 approach. To this end, we first outline the setup of the case study. Afterward, we summarize
 344 as well as discuss the correspondingly obtained results.

345 4.1 Setup

346 The basis of the case study was provided by the tool mentioned above (again, available
 347 at <https://github.com/cda-tum/rail>) based on the symbolic formulations presented in
 348 Sec. 3. This tool’s initialization requires defining a temporal discretization, i.e., a value for
 349 the time interval length Δt . Choosing the correct value of Δt compromises computational
 350 time and accuracy. Within the national railway company of Germany, Deutsche Bahn AG
 351 (DB), Δt is usually chosen to be 15 seconds (cf. [22]). While optimizing Δt when using the
 352 presented model is potentially interesting, this is out of the scope of this paper. Hence, we
 353 follow DB’s default and run all experiments with a temporal resolution of $\Delta t = 15s$.

354 As a computing device, we utilized an AMD Ryzen Threadripper PRO 5955WX system
 355 using a 4.0-4.5 GHz CPU (16 cores) and 128GB RAM running on Ubuntu 20.04. We use the
 356 C++ API of Gurobi version 10.0.1 [16] to solve the considered instances. A 1h hard timeout
 357 for solving each instance has been set.

358 We considered a range of sample train networks as benchmarks, including typical (simple)
 359 test cases (such as single tracks or stations) and real-world examples. More precisely:

- 360 ■ *Single track* considers only one line on which trains have to slow down towards the end.
 361 To enable the possibility of multiple trains following each other VSS have to be placed.
 362 We also consider a variant with a scheduled stop (i.e., station) in between.
- 363 ■ *Highspeed track* is a variant of the example above but considering a larger distance and
 364 faster trains. We consider variants with 2 and 5 trains following each other, respectively.
- 365 ■ *Simple 2-track station* is a basic station with two tracks that trains can approach from
 366 different directions.
- 367 ■ *Simple network* consists of trains that have to trespass a single track in opposite directions.
 368 For this, there is a bypass in the middle where trains do not have a scheduled stop.
- 369 ■ *Overtake* models the situation of faster trains overtaking slower trains at a bypass.
- 370 ■ *Stammstrecke* is a real-world inspired instance. For this, we model the Munich S-Bahn
 371 Stammstrecke between Pasing and Munich East using publicly available data on tracks [7],
 372 timetable [8], and technical data of the trains (DB BR 423) [20]. We reduced the train
 373 following times slightly to enforce the necessity of VSS.

374 Figures of the respective track plans and more details on the used timetables are pro-
 375 vided in Appendix A. Furthermore, all benchmarks are available through the open-source
 376 implementation at <https://github.com/cda-tum/rail>.

377 For each benchmark, the design task reviewed in Sec. 2.2 has been solved using three
 378 different degrees of detail: The base model (Sec. 3.1), this model extended by considering
 379 train dynamics (Sec. 3.2), and this model further extended by considering braking curves
 380 directly without approximation (Sec. 3.3). Additionally, we considered two cases, one in
 381 which the routes had to be determined by the tool and another in which the routes had been
 382 fixed beforehand (Sec. 3.4).

383 The number of placed VSS ($\#VSS$) is optimized for every benchmark instance. To assess
 384 the model’s efficiency, we measured the runtime (t , in CPU seconds) for creating and solving
 385 each instance. Since the solver guarantees that the optimum number of VSS will be found
 386 (within the bounds of the model’s detail), another criterion is needed to assess model accuracy.
 387 The model, including train dynamics and braking curves, is the most detailed and, thus,
 388 is, by definition, the most accurate. The less detailed models can generate solutions that,
 389 while valid within the level of detail of the formulation, cannot be mapped to reality directly.
 390 Specifically, taking the routes and VSS placements generated by less detailed models and
 391 checking them with the highest detail, it is often revealed that the routes cannot be run on

■ **Table 1** Experimental results

(a) With routing included

		Base Model			+ Train Dynamics			+ Braking Curves	
		t [s]	#VSS	Delay [s]	t [s]	#VSS	Delay [s]	t [s]	#VSS
Single Track	Without Station	53.8	7	30	53.5	8	30	290	9
	With Station	33.6	3	75	27.0	4	30	237	4
Highspeed Track	2 Trains	16.0	10	45	14.6	10	30	126	18
	5 Trains	443	10	60	604	10	30	1757	18
Simple 2-Track Station		6.5	1	15	9.2	1	0	8.5	1
Simple Network		>1h	n/a	n/a	>1h	n/a	n/a	>1h	n/a
Overtake		12.4	8	45	15.8	8	45	14.0	14
Stammstrecke	4 Trains	6.3	0	30	5.5	6	15	11.0	6
	8 Trains	17.8	0	60	15.0	14	30	68.1	14
	16 Trains	77.5	0	90	55.7	15	45	83.4	15

(b) With fixed routes

		Base Model			+ Train Dynamics			+ Braking Curves	
		t [s]	#VSS	Delay [s]	t [s]	#VSS	Delay [s]	t [s]	#VSS
Single Track	Without Station	42.0	7	45	62.5	8	30	138	9
	With Station	15.7	3	60	15.7	4	45	38.3	4
Highspeed Track	2 Trains	19.0	10	45	16.0	10	30	83.8	18
	5 Trains	536	10	45	504	10	30	1610	18
Simple 2-Track Station		0.7	1	15	0.7	1	0	1.6	1
Simple Network		64.6	5	60	41.0	5	60	1433	6
Overtake		1.4	8	45	1.5	8	45	2.7	14
Stammstrecke	4 Trains	1.2	0	45	1.1	6	15	1.3	6
	8 Trains	2.9	0	60	2.7	14	30	3.6	14
	16 Trains	7.7	0	105	6.9	15	45	13.7	15

392 the computed layout precisely as they were computed because block signaling constraints
 393 might be violated. When this safety constraint is violated, a train must halt and wait before
 394 getting the move authority. This leads to a delay (*Delay*) between the given schedule and the
 395 schedule that is possible on the computed layout in reality. We use this delay as a criterion
 396 to assess model accuracy in the following. By definition, the model including train dynamics
 397 and braking curves does not incur a delay (i.e., it is the most accurate of the considered
 398 model and serves as ground truth).

399 4.2 Results

400 The results of the conducted case study are summarized in Tab. 1. Tab. 1a provides results
 401 for instances in which the routes have *not* been fixed and, hence, have to be determined by
 402 the design method. Tab. 1b provide results for instances in which the routes have been fixed.
 403 The first columns denote the considered benchmarks as described above, while the following
 404 columns provide the number of VSS as well as the delay of the obtained solution and the
 405 runtime for each of the considered settings.

406 From the results, several interesting conclusions can be drawn. First, and most obviously,
 407 fixing the routes has a severe impact on the efficiency of the solution (comparing Tab. 1a and
 408 Tab. 1b). Fixing them beforehand reduces the search space substantially and, hence, yields
 409 substantially better runtimes. In the case of *Simple Network* it even allows one to complete
 410 the design task within the given time limit. At the same time, this sometimes is achieved
 411 at the expense of quality (as seen by the fact that some results obtained while considering

412 fixed routes have a worse delay). This can easily be explained by the fact that having routes
413 not fixed allows for a higher degree of freedom to find better solutions (causing the higher
414 runtime but may yield slightly better results). However, since the best possible routes are
415 always rather apparent in the considered benchmark, this effect only marginally affects the
416 quality of the results.

417 Besides that, the symbolic formulation’s level of detail obviously affects the accuracy
418 of the results and the efficiency of the solving process. Here, one would expect that the
419 base model (i.e., the most simplistic/abstract) model has the lowest accuracy but the best
420 efficiency, while it is vice versa for the more detailed formulations. While this is generally
421 true, interesting insights can be unveiled when checking out the numbers in detail. In fact,
422 additionally considering train dynamics in the base model hardly degrades the efficiency
423 (i.e., runtime) of the solving process (it even gets better sometimes). At the same time, the
424 solution accuracy either improves or remains equivalent. Hence, when choosing between
425 the base model and the base model plus train dynamics, the latter is the better option
426 as it provides almost the same accuracy while being more efficient. This behavior can be
427 explained by the fact that including train dynamics reduces the number of feasible train
428 movements—and, thus, the search space—without adding too much complexity.

429 More complex is the trade-off when additionally considering braking curves. This sub-
430 stantially affects the efficiency and leads to increased runtimes which are factors (sometimes
431 even magnitudes) higher than for the other models. At the same time, this provides the best
432 accuracy of all models considered in this work. This clearly shows the potential offered by
433 the proposed approach: The user can decide whether he/she wants a quick but less accurate
434 solution; or whether he/she is willing to “invest” larger computation times to get more
435 accurate solutions. The tool presented in this work allows the end user to do both.

436 **5** Conclusions & Outlook

437 In this work, we considered symbolic formulations for automatically determining ETCS HL3
438 layouts. While previous work relied on discrete formulations, we explicitly proposed continu-
439 ous formulations that are, e.g., essential to model concepts such as train dynamics or braking
440 curves properly. To this end, we introduced a *Mixed Integer Linear Programming* (MILP)
441 formulation. The resulting approach is flexible and allows users to explicitly include/exclude
442 certain constraints or simplify the model as needed. By that, he/she can decide whether a
443 fast but less accurate solution or a slow but more accurate solution should be generated.

444 A case study showcased the corresponding trade-off between accuracy and efficiency.
445 While these confirmed some obvious expectations (the less precise the model, the more
446 efficient the solving process and vice versa), some rather counter-intuitive insights are also
447 unveiled. For example, additionally considering train dynamics makes the model more precise
448 but hardly degrades (and sometimes even improves) the efficiency (i.e., runtime) of the
449 solving process. In contrast, considering braking curves substantially affects the efficiency,
450 i.e., leads to increased runtimes which are factors (sometimes even magnitudes) higher than
451 for other models.

452 The proposed methods (whose implementations are also publicly available in open-source
453 at <https://github.com/cda-tum/rail> as part of the *Munich Train Control Toolkit*, MTCT)
454 allow the users to evaluate such trade-offs. Furthermore, the resulting methods/tool has
455 been implemented in a modular and flexible fashion—allowing for easy integration of further
456 aspects in the future, such as optimizing train schedules or maximizing the throughput of
457 additional freight trains. Those developments are left for future work. Overall, the presented
458 work provides a solid basis for the design of ETCS HL3 layouts at different degrees of
459 accuracy.

460 ——— **References** ———

- 461 1 Tobias Achterberg and Eli Towle. Non-convex quadratic optimization, 2020. URL: <https://www.gurobi.com/events/non-convex-quadratic-optimization/>.
- 462
- 463 2 Maarten Bartholomeus, Laura Arenas, Roman Treydel, Francois Hausmann, Nobert Geduhn,
464 and Antoine Bossy. ERTMS Hybrid Level 3. *SIGNAL + DRAHT (110) 1+2/2018*, pages
465 15–22, 2018. URL: [https://www.eurailpress.de/fileadmin/user_upload/SD_1_2-2018_](https://www.eurailpress.de/fileadmin/user_upload/SD_1_2-2018_Bartholomaeus_ua.pdf)
466 [Bartholomaeus_ua.pdf](https://www.eurailpress.de/fileadmin/user_upload/SD_1_2-2018_Bartholomaeus_ua.pdf).
- 467 3 Ralf Borndörfer and Thomas Schlechte. Solving railway track allocation problems. In
468 *Operations Research Proceedings*, pages 117–122. Springer Berlin Heidelberg, 2008. URL:
469 https://doi.org/10.1007/978-3-540-77903-2_18.
- 470 4 Malachy Carey and Sinead Carville. Scheduling and platforming trains at busy complex
471 stations. *Transportation Research Part A: Policy and Practice*, 37(3):195–224, 2003. URL:
472 [https://doi.org/10.1016/S0965-8564\(02\)00012-5](https://doi.org/10.1016/S0965-8564(02)00012-5).
- 473 5 C.S. Chang and D. Du. Further improvement of optimisation method for mass transit signalling
474 block-layout design using differential evolution. *IEE Proceedings - Electric Power Applications*,
475 146(5):559, 1999. URL: <https://doi.org/10.1049/ip-epa:19990223>.
- 476 6 Andrea D’Ariano, Dario Pacciarelli, and Marco Pranzo. A branch and bound algorithm for
477 scheduling trains in a railway network. *European Journal of Operational Research*, 183(2):643–
478 657, 2007. URL: <https://doi.org/10.1016/j.ejor.2006.10.034>.
- 479 7 DB Netz AG. Infrastrukturregister, 2023. URL: <https://geovdbn.deutschebahn.com/isr>.
- 480 8 DB Regio AG. Fahrpläne S-Bahn München, 2023. URL: [https://www.s-bahn-muenchen.de/](https://www.s-bahn-muenchen.de/fahren/fahrplaene)
481 [fahren/fahrplaene](https://www.s-bahn-muenchen.de/fahren/fahrplaene).
- 482 9 Stefan Dillmann and Reiner Hähnle. Automated planning of ETCS tracks. In *Reliability,*
483 *Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*,
484 pages 79–90. Springer International Publishing, 2019. URL: [https://doi.org/10.1007/](https://doi.org/10.1007/978-3-030-18744-6_5)
485 [978-3-030-18744-6_5](https://doi.org/10.1007/978-3-030-18744-6_5).
- 486 10 EEIG ERTMS Users Group. ERTMS/ETCS hybrid train detection. Technical Report 16E042,
487 2022. Version 1E. URL: [https://ertms.be/wp-content/uploads/2023/06/16E0421F_HTD.](https://ertms.be/wp-content/uploads/2023/06/16E0421F_HTD.pdf)
488 [pdf](https://ertms.be/wp-content/uploads/2023/06/16E0421F_HTD.pdf).
- 489 11 Stefan Engels, Tom Peham, Judith Przigoda, Nils Przigoda, and Robert Wille. Design tasks
490 and their complexity for Hybrid Level 3 of the European Train Control System. 2023. URL:
491 <https://doi.org/10.48550/arXiv.2308.02572>.
- 492 12 European Union Agency for Railways. Introduction to ETCS braking curves. Technical
493 Report ERA_ERTMS_040026, 2020. Version 1.5. URL: [https://www.era.europa.eu/](https://www.era.europa.eu/system/files/2022-11/IntroductiontoETCSbrakingcurves.pdf)
494 [system/files/2022-11/IntroductiontoETCSbrakingcurves.pdf](https://www.era.europa.eu/system/files/2022-11/IntroductiontoETCSbrakingcurves.pdf).
- 495 13 O. Gemine, A. Hougardy, and E. Lepailleur. ERTMS unit: Assignment of values to
496 ETCS variables. Technical Report ERA_ERTMS_040001, European Union Agency for
497 Railways, 2023. Version 1.33. URL: [https://www.era.europa.eu/system/files/2023-02/](https://www.era.europa.eu/system/files/2023-02/ETCSvariablesandvalues.pdf)
498 [ETCSvariablesandvalues.pdf](https://www.era.europa.eu/system/files/2023-02/ETCSvariablesandvalues.pdf).
- 499 14 D.C. Gill and C.J. Goodman. Computer-based optimisation techniques for mass transit railway
500 signalling design. *IEE Proceedings B Electric Power Applications*, 139(3):261, 1992. URL:
501 <https://doi.org/10.1049/ip-b.1992.0031>.
- 502 15 Jan-Willem Goossens, Stan van Hoesel, and Leo Kroon. On solving multi-type railway line
503 planning problems. *European Journal of Operational Research*, 168(2):403–424, 2006. URL:
504 <https://doi.org/10.1016/j.ejor.2004.04.036>.
- 505 16 Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. URL: [https://www.](https://www.gurobi.com)
506 [gurobi.com](https://www.gurobi.com).
- 507 17 Lylly Hernández and Sascha Hardel. Section breaks and level crossings limit capacity increases
508 under ETCS Level 2. *SIGNAL + DRAHT (115) 1+2 / 2023*, pages 24–30, 2023.
- 509 18 B.R. Ke and N. Chen. Signalling blocklayout and strategy of train operation for saving energy
510 in mass rapid transit systems. *IEE Proceedings - Electric Power Applications*, 152(2):129,
511 2005. URL: <https://doi.org/10.1049/ip-epa:20045188>.
- 512 19 Torsten Klug, Markus Reuther, and Thomas Schlechte. Does laziness pay off? - a lazy-
513 constraint approach to timetabling. In Mattia D’Emidio and Niels Lindner, editors, *22nd*
514 *Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and*

- 515 *Systems (ATMOS 2022)*, volume 106. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.
516 URL: <https://doi.org/10.4230/OASICS.ATMOS.2022.11>.
- 517 20 Thomas Künzel. *Triebwagen für den zukünftigen Nah- und Regionalverkehr in Deutsch-*
518 *land*. PhD thesis, TU Berlin, 2019. URL: [https://depositonce.tu-berlin.de/items/](https://depositonce.tu-berlin.de/items/76de5c51-6cad-4250-abd5-df7b35643409)
519 [76de5c51-6cad-4250-abd5-df7b35643409](https://depositonce.tu-berlin.de/items/76de5c51-6cad-4250-abd5-df7b35643409).
- 520 21 Richard M. Lusby, Jesper Larsen, Matthias Ehrgott, and David Ryan. Railway track allocation:
521 models and methods. *OR Spectrum*, 33(4):843–883, dec 2009. URL: [https://doi.org/10.](https://doi.org/10.1007/s00291-009-0189-0)
522 [1007/s00291-009-0189-0](https://doi.org/10.1007/s00291-009-0189-0).
- 523 22 Richard M. Lusby, Jesper Larsen, Matthias Ehrgott, and David M. Ryan. A set packing
524 inspired method for real-time junction train routing. *Computers & Operations Research*,
525 40(3):713–724, 2013. URL: <https://doi.org/10.1016/j.cor.2011.12.004>.
- 526 23 Bjørnar Luteberget. *Automated Reasoning for Planning Railway Infrastructure*. PhD thesis,
527 Univ. of Oslo, May 2019. URL: [https://www.mn.uio.no/ifi/english/research/projects/](https://www.mn.uio.no/ifi/english/research/projects/railcons/documents/luteberget-thesis-b5-2019-09-17.pdf)
528 [railcons/documents/luteberget-thesis-b5-2019-09-17.pdf](https://www.mn.uio.no/ifi/english/research/projects/railcons/documents/luteberget-thesis-b5-2019-09-17.pdf).
- 529 24 Richard Oberdieck, Kostja Siefen, Jaromil Najman, and Ed Klotz. Tech talk - a practi-
530 cal tour through non-convex optimization, 2021. URL: [https://www.gurobi.com/events/](https://www.gurobi.com/events/tech-talk-a-practical-tour-through-non-convex-optimization/)
531 [tech-talk-a-practical-tour-through-non-convex-optimization/](https://www.gurobi.com/events/tech-talk-a-practical-tour-through-non-convex-optimization/).
- 532 25 Jörn Pachl. *Railway Signalling Principles: Edition 2.0*. Universitätsbibliothek Braunschweig,
533 2021. URL: <https://doi.org/10.24355/dbbs.084-202110181429-0>.
- 534 26 Tom Peham, Judith Przigoda, Nils Przigoda, and Robert Wille. Optimal railway routing
535 using virtual subsections. In *Reliability, Safety, and Security of Railway Systems. Modelling,*
536 *Analysis, Verification, and Certification*, pages 63–79. Springer International Publishing, 2022.
537 URL: https://doi.org/10.1007/978-3-031-05814-1_5.
- 538 27 Vahid Ranjbar, Nils O.E. Olsson, and Hans Sipilä. Impact of signalling system on capacity –
539 comparing legacy ATC, ETCS Level 2 and ETCS Hybrid Level 3 systems. *Journal of Rail*
540 *Transport Planning & Management*, 23:100322, 2022. URL: [https://doi.org/10.1016/j.](https://doi.org/10.1016/j.jrtpm.2022.100322)
541 [jrtpm.2022.100322](https://doi.org/10.1016/j.jrtpm.2022.100322).
- 542 28 Thomas Schlechte, Ralf Borndörfer, Jonas Denißen, Simon Heller, Torsten Klug, Michael
543 Küpper, Niels Lindner, Markus Reuther, Andreas Söhlke, and William Steadman. Timetable
544 optimization for a moving block system. *Journal of Rail Transport Planning & Management*,
545 22:100315, 2022. URL: <https://doi.org/10.1016/j.jrtpm.2022.100315>.
- 546 29 Lars Schnieder. *Communications-Based Train Control (CBTC)*. Springer Berlin Heidelberg,
547 March 2021. URL: <https://doi.org/10.1007/978-3-662-62876-8>.
- 548 30 Lars Schnieder. *European Train Control System (ETCS)*. Springer Berlin Heidelberg, 2021.
549 URL: <https://doi.org/10.1007/978-3-662-62878-2>.
- 550 31 Valeria Vignali, Federico Cuppi, Claudio Lantieri, Nicola Dimola, Tomaso Galasso, and
551 Luca Rapagnà. A methodology for the design of sections block length on ETCS L2 railway
552 networks. *Journal of Rail Transport Planning & Management*, 13:100160, 2020. URL:
553 <https://doi.org/10.1016/j.jrtpm.2019.100160>.
- 554 32 Robert Wille, Tom Peham, Judith Przigoda, and Nils Przigoda. Towards automatic design and
555 verification for Level 3 of the European Train Control System. In *2021 Design, Automation &*
556 *Test in Europe Conference & Exhibition (DATE)*. IEEE, 2021. URL: [https://doi.org/10.](https://doi.org/10.23919/date51398.2021.9473935)
557 [23919/date51398.2021.9473935](https://doi.org/10.23919/date51398.2021.9473935).

558 **A** Benchmarks

559 In the following we present the track layouts (all referenced figures can be found at the end
560 of the appendix) used for the case study together with brief descriptions on the timetable.
561 For detailed information on timetables, we refer to the sample instances provided through
562 our open-source implementation.

563 **Single Track**

564 *Single Track Without Station* is a single track that is 15km long with no TTD. Two trains
565 follow each other with a 1 minute headway.

566 The track layout of *Single Track With Station* is shown in Fig. 6. Three trains travel
 567 from A to B with 90 seconds separation. The first two trains have a scheduled stop at S_1 .

568 Highspeed Track

569 *Highspeed Track* is a single track that is 50km long with no TTD. Two to five trains are
 570 following each other with a 1 to 2 minute headway while slowing down towards the end.

571 Simple 2-Track Station

572 The track layout is given in Fig. 7. Two trains are moving from left to right, one (longer)
 573 train from right to left. All three trains have a scheduled stop in S_1 at more or less the same
 574 time.

575 Simple Network

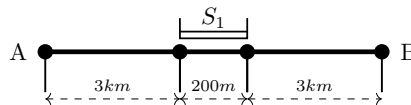
576 The track layout is given in Fig. 8. Two slow trains move from left to right and right to left
 577 respectively with two scheduled stops. Faster trains follow on the main line without stopping.

578 Overtake

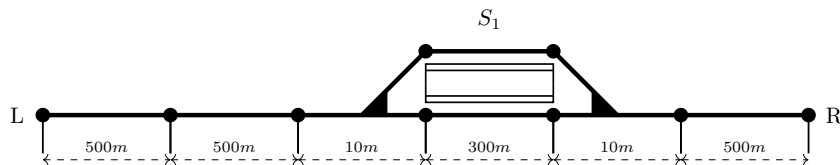
579 The track layout is given in Fig. 9. Three trains enter at A and leave at B in the reverse
 580 order, hence, have to overtake each other. Additionally, the first train has a scheduled stop
 581 in S_1 .

582 Stammstrecke

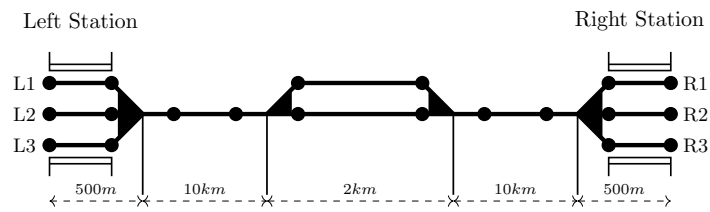
583 We model the Munich S-Bahn Stammstrecke between Pasing and Munich East using publicly
 584 available data on tracks [7]. The track layout is shown in Fig. 10. S-Bahn trains in Munich
 585 are mainly of type *DB BR 423* with technical data described in [20]. The timetable is inspired
 586 by [8] and consists of trains entering at Pasing, Laim or Donnersbergerbrücke traveling to
 587 Munich East and vice versa in the opposite direction. Depending on the instance, 2, 4, and 8
 588 trains (per direction) were considered following each other approximately every 90 seconds.



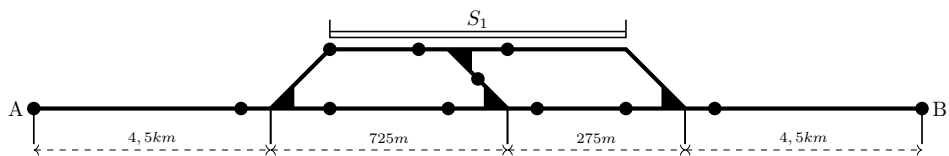
■ **Figure 6** Tracks of *Single Track With Station*



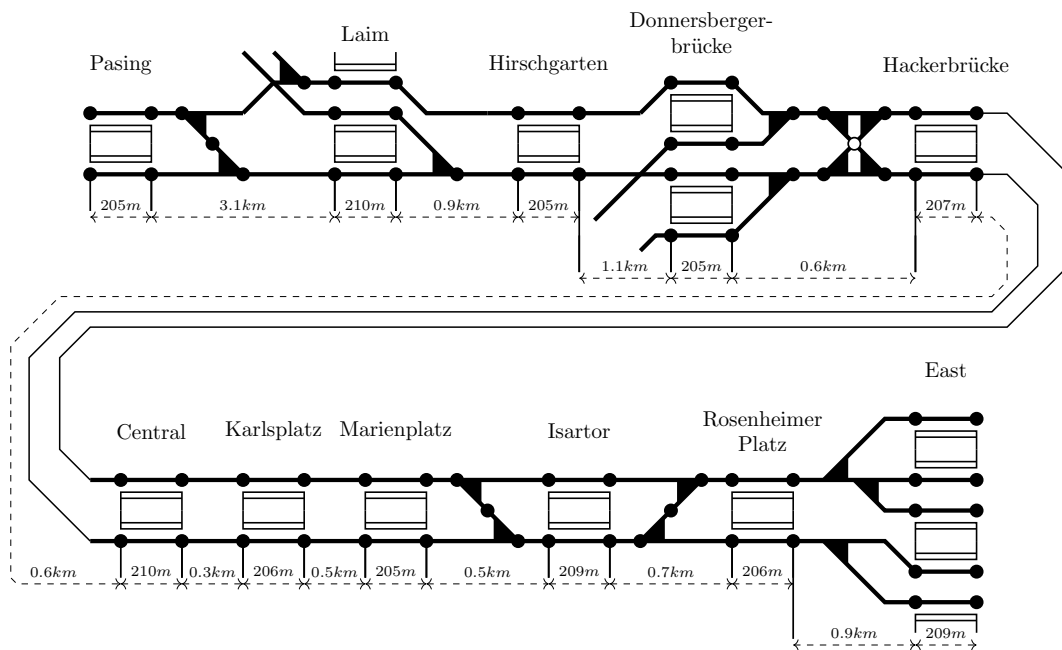
■ **Figure 7** Tracks of *Simple 2-track station*



■ Figure 8 Simple Network



■ Figure 9 Overtake



■ Figure 10 Stammstrecke (Munich S-Bahn)