TRAIN MOVEMENT DYNAMICS WITHIN ANYLOGIC TOOL

Roman Diviš[(a)], Antonín Kavička[(b)]

[(a)] Faculty of Electrical Engineering and Informatics, University of Pardubice
[(b)] Faculty of Electrical Engineering and Informatics, University of Pardubice

[(a)]roman.divis@student.upce.cz, [(b)]antonin.kavicka@upce.cz

ABSTRACT
The AnyLogic simulation tool serves for the simulation of railway traffic. The dynamics of train motion, however, are modelled in a simplified way based on acceleration invariables. When used for industrial purposes, such calculations can be inaccurate and the simulation thus does not yield valid results. In order to make the calculations of dynamics more accurate, the TrainDyn Library was created. It implements the train movement dynamics according to reality and includes the impact of forces acting during the train´s motion.

Keywords: train movement dynamics, railway traffic simulation, AnyLogic tool

1. ANYLOGIC

The *AnyLogic* simulation represents a highly universal modelling tool enabling the utilisation of discreet simulation methods and agent-oriented approach. The creation of simulation models is supported by component libraries, which include various objects for the simulation of technological processes and social or economic systems. (Borschev 2013)

1.1. Enteprise Library

The *Enterprise* library represents an elementary object library for the creation of discreet simulation. The simulation model creator can use components for the creation of entities, operational processes and decision-making logic.

The *AnyLogic* simulation tool can be very well adjusted to the needs of a particular simulation. Utilisation of the *Java* programming language gives the possibility to influence the behaviour of the whole simulation. Individual library objects include events (e.g. creating an entity, processing an entity within a particular object, etc.) that enable to execute one´s own *Java* code. The simulation model creator thus has the possibility to create his/her own entity classes or auxiliary classes and methods to be used within the simulation model. It is then to some extent possible to influence the behaviour of individual objects using the available *API* of the *AnyLogic* tool.

Apart from the *Enterprise library*, it is also possible to use objects from other libraries that can be arbitrarily connected, which enables the creation of a complex simulation model.

1.2. Rail Library

AnyLogic also includes a library for the simulation of rail traffic. The library allow to propose and create railway traffic simulation within the *AnyLogic* environment. The simulation enables the creation of trains, their placement within the railway infrastructure and the management of their movement on the line.

The library´s basic element, *TrainMoveTo*, serves for the setup of the train movement. Either the train has a set destination and the journey is automatically calculated or the train travels freely according to the set-up points on the track. By using other parameters it is possible to set up the acceleration or deceleration prior to the termination of the train´s journey.

The dynamics of the individual trains´ movement are, however, modelled in a simplified way on the basis of invariables defining the train´s acceleration or deceleration. Many factors influencing the train´s movement are not included. Majority of the simulation studies focusing on the microscopic simulation of the railway traffic, however, require a precise simulation of the rolling stock´s movement. Precision may be crucial especially in industrial utilisation so that the yielded results can be considered valid. Therefore, it is necessary to adjust the simulation model and add an alternative way of dealing with the train movement dynamics.

2. TRAIN MOVEMENT DYNAMICS

Describing the movement of a system of objects requires the definition of the system´s state at each moment. Mechanical system can be replaced by a point mass. Point mass is a particle, whose dimensions are negligible in relation to the other parameters. Phase coordinates of the point mass define its position within the three-dimensional space and its velocity at each moment. (Garg 1984)

Changes of phase coordinates are usually described by means of differential equations. A free object of n degrees of freedom requires n differential equations for the description of its movement in space. When a system consists of m mutually interconnected objects, it is necessary to set $m.n$ differential equations.

Describing each railway vehicle in the system would be rather problematic. Experiments showed that a sufficiently precise calculation can be achieved by making a number of assumptions.

The rolling stock´s movement can be mathematically described as the movement of a point mass with one degree of freedom. When exploring the movement of the rolling stock, it is possible to consider only the external forces, which act along the movement. The movement can be calculated according to the input and output without considering the dynamics of the processes taking place within the system. To simplify the calculation of the movement´s dynamics it is further possible to use linear approximation of some non-linear continuous functions of the acting forces.

2.1. Mathematic Model

On the basis of the previous assumptions, the increase of work of the external forces can be defined as:

$$dA = (F - B - O).dL \tag{1}$$

in which F is the traction force, B breaking force, O resistance force and dL represents the gain in distance. Further adjustment of the equation requires the use of the relationship for the calculation of kinetic energy during translation and rotation:

$$E = \frac{1}{2} \cdot m \cdot v^2 + \sum \frac{1}{2} \cdot I \cdot \omega^2 \tag{2}$$

in which I is a moment of inertia, ω is angular velocity, m weight of the system and v velocity of the system. The following equation holds for the moment of inertia:

$$I = m_r \cdot i^2 \tag{3}$$

where m_r represents the weight of the rotating parts and i is the distance from the rotation axis. The following equation holds for the angular velocity:

$$\omega = v \cdot \frac{u}{R} \tag{4}$$

where R is the wheel radius and u the velocity ratio of the wheel´s revolution and the respective rotating part. After the equations are entered into the formula for the calculation of kinetic energy, kinetic energy is expressed as:

$$E = \frac{m.v^2.\rho}{2} \tag{5}$$

where ρ is the coefficient of the rotating parts in:

$$\rho = 1 + \frac{\sum m_r.i^2.u^2}{m.R^2} \tag{6}$$

The change of kinetic energy under any change of state can be expressed in a differential form as:

$$dE = m.\rho.v.dv \tag{7}$$

The work of the force acting on the system during a particular route equals the increase of the system´s kinetic energy. Thus, dE = dA, and therefore, after entering into the previous equations (1,7):

$$m.\rho.v.dv = (F - B - O).dL \tag{8}$$

The formula dL = v.dT, in which dT is a gain in time, is valid for the gain in distance. After entering into the equation and the necessary adjustments, we arrive at the resulting formula:

$$F - B - O = m \cdot \rho \cdot \frac{dv}{dT} \tag{9}$$

2.2. Analysing the Acting Forces

The following section describes the individual forces acting in the movement of the rolling stock and their possible calculations and modelling for the purposes of computer simulation. (Iwnicki 2006, Dukkipati 2000)

2.2.1. Resistance Forces

Resistance forces represent a group of forces acting against the motion of vehicles, with the exception of braking forces. On the whole, resistance forces can be referred to as rolling resistance. It can be divided into two groups (i) vehicle resistance and (ii) track resistance.

Vehicle resistance depends on the construction of each individual vehicle. The physical basis of vehicle resistance rests in the point of contact of the vehicle with the track and the external environment. Vehicle resistance includes (i) resistance of the bearings, (ii) rolling resistance, and (iii) air resistance. Individual resistance forces can be expressed by means of explicit formulas, but their practical calculations are rather complex. Practical calculations make use of equations for calculating the coefficient of vehicle resistance instead of the specific value of vehicle resistance. The coefficient of vehicle resistance is defined as the quotient of vehicle resistance and the vehicle´s weight. The coefficient of vehicle resistance can be expressed as a degree two polynomial with the velocity of the vehicle´s movement as a variable:

$$o_v = a + b.v + c.v^2 \tag{10}$$

the polynomial´s coefficients a, b, c were gained by empirical measurements and are available in the vehicle´s technical documentation.

Resistance of the track depends on the track´s state and construction. It consists of (i) inclination resistance, (ii) curve resistance and (iii) tunnel resistance.

Inclination resistance comes into play during the uphill or downhill journey of the rolling stock. Such movement incites a component of the gravitational force which is parallel to the track´s inclination. During an uphill movement it creates resistance against the

acting traction force. Inclination resistance can be calculated as:

$$O_{skl} = G.\sin \propto \quad (11)$$

where G is the weight of the vehicle and α is the angle of ascension or descending. The angle of ascension or descending is not used in practice; instead there are measurements of height gained during one-thousand-metre long sections of the vehicle´s journey. Such gain in height is referred to as inclination and is defined as:

$$s = 10^3 \tan \propto \quad (12)$$

In railway traffic, the value of the track inclination angle is very small (around 4 °). With small angles, the difference between functions tan α and sin α is rather small, which allows the use of $\tan \propto = \sin \propto$ and we can proceed to:

$$O_{skl} = G.\sin \propto = G.\tan \propto = \frac{G.s}{10^3} \quad (13)$$

The coefficient of inclination resistance can thus be expressed as:

$$o_{skl} = \frac{10^3.O_{skl}}{G} = \frac{10^3.s}{10^3} = s \quad (14)$$

Curve resistance occurs during the vehicle´s journey through a curve during which the friction between the track and the vehicle´s wheels increases. The general function for the calculation of the curve resistance coefficient is rather complex and involves a number of factors (track gauge, skid friction, track´s degree of incline, curve radius, etc.). Due to the complexity of the issue, practical calculations use empirical formula such as:

$$o_{obl} = \frac{a}{R-b} \quad (15)$$

with R as the curve radius and a, b as the defined coefficients for the given track type.

Tunnel resistance is caused by the air inside a tunnel. It is caused by the increase of drag during the vehicle´s passage through a tunnel. The vehicle pushes an air column which is unable to escape and it causes a massive whirl. The tunnel resistance coefficient has its defined values of 2 $N.kN^{-1}$ for a single-track tunnel and 1 $N.kN^{-1}$ for a double-track tunnel.

2.2.2. Braking Forces

Braking force acts against the direction of the vehicle´s movement in a similar way as the resistance forces. Contrary to the resistance forces thought, braking force is created artificially with the aim of slowing down or halting the vehicle. Mechanical brakes use the friction resistance. The calculation of the coefficient of friction caused by the braking process can be based on the force, created by the brake on a wheel, and the vehicle´s velocity. Practical calculations can assume that braking causes a constant deceleration of vehicles.

2.2.3. Traction Forces

Traction force acts in the direction of the vehicle´s movement and is caused by a tractor. Traction force can be further divided according to the point of action into (i) indicated traction force, (ii) traction force on the wheel circumference and (iii) traction force on the coupling. As to the calculation of dynamics, we are most interested in the traction force on the wheel circumference, which represents the indicated traction force with the deduction of losses within the engine and losses caused by the transmission of force from the engine to the wheels.

The extent of the traction force, which the tractor can cause, can change according to operational velocity. The dependence between the traction force and velocity under specific output of the tractor cannot be expressed analytically. It is thus expressed graphically in a square grid system and is referred to as traction characteristics.

2.3. Movement Equation of the Rolling Stock

The previous analysis of the individual forces acting during the vehicle´s journey enables the formulation of the differential equation for tractors as well as pulled vehicles:

$$f_O(v) - b(v) - o_{vh}(v) - o_t(v) = \rho \cdot \frac{10^3}{g} \cdot \frac{dv}{dt} \quad (16)$$

where f_O is the specific traction force on the wheel circumference, b specific braking force, o_{vl} the coefficient of vehicle resistance of the train, o_t the coefficient of track resistance, ρ the coefficient of rotating parts, v velocity and t time.

Solving the differential equation on computer requires one of the numerical methods. The simplest is the Euler numerical method, which solves the calculation in steps and the approximation derivation is calculated as a function value in each respective step.

In order to find out the dependence between the velocity and the distance within time T_0 to T_n, the initial velocity of the movement V_0 must be known. The equation can be expressed as:

$$k.\frac{dv}{dt} = \varphi(v,t) \quad (17)$$

in which the right side of the equation is considered a constant in each iteration step. Time T_0 to T_n is divided into n equal sections, the calculation step h is then defined as:

$$h = \frac{T_n - T_0}{n} = \Delta T \quad (18)$$

Derivation in each point of the curve $v(t)$ is substituted by the quotient of final differences and the equation becomes:

$$k.\frac{\Delta v}{\Delta t} = \varphi(v,t) \quad (19)$$

and for each step it is valid that:

$$k.\Delta v_0 = k.(V_1 - V_0) = \varphi(T_0, V_0).h \text{ for v(T0)} = \text{V0}$$
$$k.\Delta v_1 = k.(V_2 - V_1) = \varphi(T_1, V_1).h \text{ for v(T1)} = \text{V1}$$
$$\ldots$$
$$k.\Delta v_n = k.(V_n - V_{n-1}) = \varphi(T_{n-1}, V_{n-1}).h \text{ for v(Tn} - 1) = \text{Vn} - 1 \quad (20)$$

Thus the equations for the calculation of velocity and covered distance in step *i*. can be expressed as:

$$V_i = V_{i-1} + \frac{h.(f_0(V_{i-1}) - b(V_{i-1}) - o_{vl}(V_{i-1}) - o_t(L_{i-1}))}{k} \quad (21)$$

$$L_i = L_{i-1} + \frac{V_{i-1}+V_i}{2}.h \quad (22)$$

The presented formulae enable the calculations of velocity and covered distance at each moment. The calculation´s precision depends on the size of the time step ΔT.

3. TRAINDYN LIBRARY

To enable the simulation of train movement using dynamics, it is necessary to adjust the simulation model. On the other hand, it would be demanding and ineffective to adjust each model and complete it with dynamics calculations. The *AnyLogic* environment allows the creation of a simulation model that can be further used through other simulation models as a library. The *TrainDyn* library was created in the same way and includes the implementation of the train journey dynamics, as described in the previous chapter.

The library implements its own calculation core, which lists detailed information on railway traffic and individual trains within the simulation model. Such information is then processed and used for the calculation of train movement dynamics.

The library uses the principle of activity scanning. The calculation of dynamics sets off periodically and the newly calculated velocity are set up for the respective train entities. This connection is carried out by the methods on the *API* interface of *AnyLogic*. The simulation model creator can adjust the sampling period as required.

The advantage of this solution is its rather simple realisation for the simulation model creator and the possibility to change the precision/complexity of calculation according to the selected sampling period. The disadvantage is that the train´s velocity between two runs of the dynamic calculations remains constant. Another shortcoming of the *AnyLogic* tool is the inability to stop the train completely. It is possible to decrease the velocity to a very small value, but a complete halt of the train is achievable only by means of a well set up *TrainMoveTo* feature.

3.1. Structures of the Railway and Trains

To carry out the dynamics calculations it is necessary to have more information about the train, which do not belong to the standard *Train* class within the *AnyLogic* tool. Information about the train´s characteristics is contained in the *TrainCharasteristics* class and includes the weight of the tractors as well as pulled vehicles, braking deceleration, traction characteristics of the locomotive, coefficients for the calculation of vehicle resistance and maximum train velocity.

To enable the work with train entities, which include the above-specified information, class *DynTrain* was created. It inherits from the standard *Train* class, which is an entity used in railway simulations. *DynTrain* includes train characteristics and other parameters needed for dynamics calculations (time lapsed from the last carried out calculation, required velocity and state of movement).

Railway traffic also lacks the parameters needed for dynamics calculations. It concerns mainly the track inclinations, location and radius of curves and location of tunnels. These parameters are passed on within the initialisation code of the simulation directly into the simulation core of the library. The simulation core is realised by the *TrainDynamics* class, which uses the *Singleton* design pattern. Therefore, a class instance can be gained from any part of the model and can be communicated with.

3.2. API Interface of the TrainDyn Library

The classes *DynTrain* and *TrainDynamics* represent the basic functional blocks for the library´s functioning. Their communication with the simulation model is carried out by means of a simple API interface.

The *DynTrain* class contains the following public methods:

- setTrainCharacteristics(TrainCharacteristics tch) – a method which sets up the train characteristics (parameters needed for the journey dynamics calculation);
- accelerate(double speed) – sets up new required velocity of the train and the dynamics calculation starts to accelerate or decelerate the train as required;
- stop() – sets up the braking of the train until a complete halt.

The *TrainDynamics* class includes the following public methods:

- static getInstance() – returns the instance of the *TrainDynamics* class (design pattern *Singleton*);
- addArc(ShapePolyLine polyline, int radius) – marks the information on a curve on the line located on the *polyline* section with the radius of *radius* metres;
- addArc(ShapePolyLine[] polyline, int radius) – similar to the previous method for the collective information adding;
- addTrain(DynTrain train) – adds the *train* to the dynamics calculation;
- removeTrain(DynTrain train) – removes the *train* from the dynamics calculation;

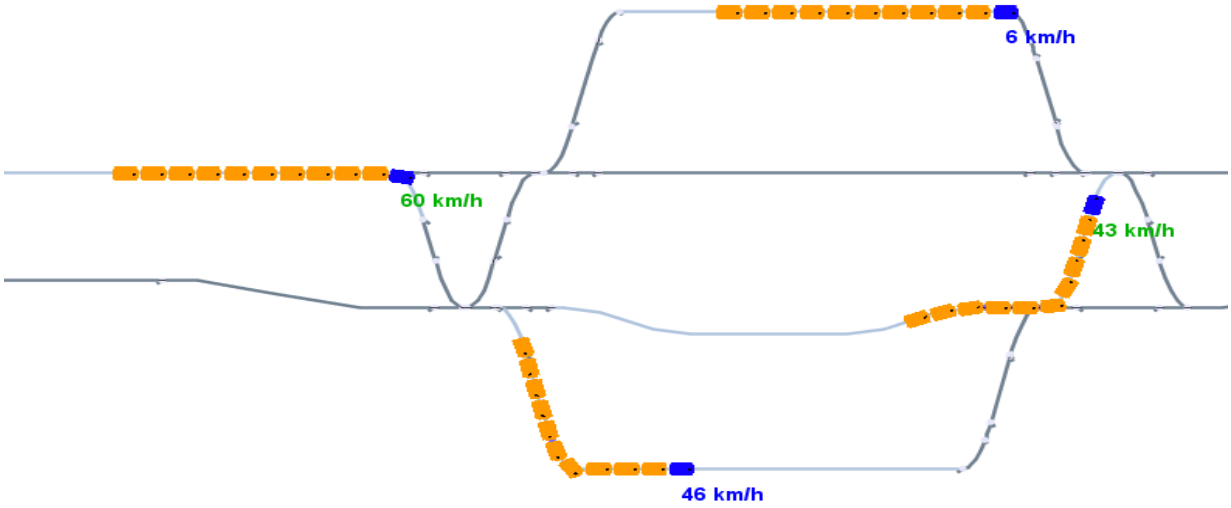

Figure 1: Testing topology simulation

- calculateSpeeds(double time) – carries out the dynamics calculation for all added trains, the *time* value provides the actual simulation time;
- processTrains(ActionTrain actionTrain) – allows for the *actionTrain* callback for all added trains within the calculation core.

The above-mentioned API enables the communication with the library and the use of its possibilities for managing train dynamics. The processes necessary for the library´s utilisation are described in the following section of this article.

3.3. Using the Library

To be able to calculate the train´s movement dynamics using the library, the simulation model creator must take several basic steps in preparing the model and then connecting the library with the simulation model. The following description presents a proposed model including the railway infrastructure and the *TrainSource*, *TrainMoveTo* and *TrainDispose* features for managing the simulation.

Prior to its use, the library must be filled into the simulation model. In the simulation´s model set-up in the *Dependencies* section, the creator must add a reference to the JAR file of the dynamics library and also set up the elementary time unit of the simulation to one second. The following step involves adding information about the railway infrastructure. The *Startup code* of the simulation model must be completed by an *addArc()* call and the other railway parameters must be set up.

Then it is necessary to adjust the *TrainSource* features for creating the entities of *DynTrain* type (or the related class) and add the method call *addTrain()*in the *On exit* code. In a similar way, the individual trains must be removed within the *TrainDispose* features, where the method call *removeTrain()* is added into the event *On enter*.

The actual calculation is realised on the basis of a periodical method call *calculateSpeeds()*. Therefore, the *Event* feature must be created in the simulation model and must be set up to the *Trigger type: Timeout* a *Mode: Cyclic*. The time of repeating the event defines the precision, but also complexity of the calculation. The starting value can be set up to one second and then adjusted as required. Action must be completed with the actual call: *TrainDynamics.getInstance().calculateSpeeds(time());*

For train movement, the features *TrainMoveTo* and *TrainSource* can be completed with a call *accelerate()/stop()*, which sets up a new mode of train movement.

3.4. Case Study

Some functionality tests and validation of the calculation results were carried out already during the implementation of the TrainDyn library. Validation was carried out in form of verification of the actual tachograph records and their comparison with the calculations.

A case study was compiled during the library´s first testing process. It includes the infrastructure of three simplified train stations. One-direction train traffic was generated within this infrastructure with a stop on a randomly chosen track at the first station. The topology is presented in figure 1. The trains are marked with their current velocity. Blue colour means that the train is braking to stop at the station, green means acceleration or constant speed. Displaying the description of each train is based on the *processTrains()* call, which moves the description to the front of the train and updates the train´s velocity.

4. CONCLUSION

In order to use the train movement dynamics calculation for the purposes of industrial railway simulations it was necessary to adjust the *AnyLogic* simulation tool. That

was the reason for designing and implementing the *TrainDyn* library which realises its own dynamics calculation core and enables the simulation model creator rather simple connection of the library with new and existing standard models. The user sets up the movement of individual trains through several basic *API* calls and the actual dynamics calculations are carried out by the library.

Currently, the library just needs the implementation of several values of track resistance, which are not yet included in the calculations. In case higher precision is required, the sampling period of dynamics calculation can be adjusted or alternative way of numerical integration can be implemented. The testing and validation of results with the current method yielded satisfactory results and thus, the implementation of more complex methods was not necessary.

REFERENCES

Garg, V., 1984. *Dynamics of Railway Vehicle Systems*. Burlington: Elsevier Science.

Iwnicki, S., 2006. *Handbook of railway vehicle dynamics*. Boca Raton: CRC/Taylor.

Borshchev, A., 2013. *The Big Book of Simulation Modeling. Multimethod Modeling with Anylogic 6*. Lisle, IL: AnyLogic North America.

Dukkipati, R.V., 2000. *Vehicle dynamics*. Boca Raton: Narosa Publishing House.