

Vasúti vontatójárművek valós adatokon alapuló egyszerűsített mozgásainak vizsgálata SUMO szimulációs környezetben

A nyílt forráskódú és szabad hozzáférésű szimulációs program adta
lehetőségek vizsgálata, és a hasznosítás lehetőségeinek meghatározása

Tudományos Diákköri Munka

Nagy Krisztián

Széchenyi István Egyetem
Építész-, Építő- és Közlekedésmérnöki Kar

Győr

2024. április

Tartalomjegyzék

Absztrakt	4
Bevezetés.....	5
1. A szimulációk.....	6
1.1 A nyílt forráskódú programok irányzata.....	7
1.2 Közlekedési szimulációs rendszerek	7
2. A vontatójármű és jellemző paraméterei a szimuláció szempontjából.....	14
2.1 A vontatójármű alaptulajdonságai	15
3. A vonóerő- és villamosfékerő görbe leképezése.....	19
3.1 A vonóerőgörbe leképezése.....	19
3.2 A villamosfékerő görbe leképezése	21
3.3 Az ellenállások	24
3.4 A vonó- és villamosfékerő görbe felhasználása	25
4. A szimulációs környezet felépítése	26
4.1 Futtatási környezet létrehozása.....	26
5. A szimuláció beállítása és specifikálása.....	28
5.1 TraCI interfész létrehozása	28
5.2 Vonóerőgörbe Python függvénybe foglalása	29
5.3 Villamosfékerő görbe Python függvénybe foglalása.....	30
5.4 A gyorsulás és lassulás finomhangolása.....	32
5.5 A drive_control eljárás.....	34
5.6 Statikus forgatókönyvi elemek meghatározása	36
5.7 Mozdonyvezetői attitűd megváltoztatása	37
5.8 Szabályzók állásainak dinamikus megváltoztatása	38
6. A szimulációval elvégzett vizsgálatok	39
6.1 Gyorsulással és lassulással végzett szimuláció	39
6.2 Összehasonlítás egy valós példával.....	45

7. Lehetőségek a továbbfejlesztésre	48
Ábrajegyzék	49
Irodalomjegyzék.....	50

Absztrakt

A szimulációs környezetekben jól jellemezhetők a járműmozgások a valós forgalom zavarása nélkül. Dolgozatom célja megvizsgálni a SUMO nyílt forráskódú közlekedési szimulációs környezet használhatóságát vasúti járművek valós járműadatokon alapuló egyszerűsített mozgásainak meghatározásához. A zárt szimulációs környezetekben általában nem, vagy csak nagyon kis mértékben lehet a mozdonyvezetői attitűdöt különböző feltételek szerint szabályozni, illetve a jármű mozgására ható paramétereket figyelembe venni és ezek alapján utasításokat adni. Dolgozatomban a felépített környezet futtatása során lehetőség nyílik a mozdonyvezetői magatartás megváltoztatására, így megvizsgálhatóvá válnak a vezetési stílus hatásai a jármű egyszerűsített mozgására. A környezet felépítése során a Siemens gyártmányú ES64U2 típusjelölésű Europrinter (ÖBB 1116, MÁV 1047), ismertebb nevén Taurus négytengelyes villamosvontatású jármű menetjellemzőivel és alapadataival határoztam meg az alapértelmezett járműtulajdonságokat. A modell előállítása során foglalkozom a vontatójármű vonóerőgörbéjének értelmezésével, és leképezhetőségének lehetőségeivel. A szimuláció során több vonattípust is megvizsgálva részletezem, hogy milyen mozgási eltérések tapasztalhatók az egyes típusok között. A dolgozatom taglalja továbbá, hogy milyen módszerrel lehet a meglévő vasúti vontatójárművet a szimulációs modell számára értelmezhető módon leképezni, és a szimuláció futtatása után keletkezett adatokat hogyan lehet tovább hasznosítani.

Kulcsszavak: szimuláció, SUMO, mozdonyvezetői attitűd, vezetési stílus, vonóerőgörbe, ES64U2, Siemens Taurus, egyszerűsített vonatmozgás

Bevezetés

Az utóbbi időben az ingadozó villamosenergia- és gázolaj áraknak köszönhetően nagyobb figyelmet kaphatnak a különféle energiahatékony járművezetésre irányuló képzések és megoldások a közlekedési ágazatokban. Ilyen képzésnek nevezhető például a mozdonyvezetők számára létrehozott ecodriving tanfolyam, ahol az alaptanfolyamokon megismert tananyag újrafeldolgozása történik meg a gazdaságos vezetésre vonatkozó fejezetek kihangsúlyozásával.

A közúti és a vasúti közlekedés esetén is ismert megoldás a járművezetők ösztönzése az energiahatékony vezetésre. Ennek értelmében a járművezetőnek a mindenkori biztonsági tényezők előtérbe helyezése mellett a gazdaságos vezetési stílusra is érdemes törekednie, hiszen a többlet odafigyelésért és az ezt eredményező megtakarított energiáért külön juttatásban is részesülhet.

A gyakornoki pozícióból adódóan jelenleg is foglalkozom a képzés utóhatásainak- és a valós körülmények közötti összefüggések eredményének vizsgálatával. Ezen vizsgálatok végzése során érzékeltem, hogy az eredmények megismeréséhez összetett helyzetek elemzésére, pontos lehatárolásokra és számos olyan egyéb ismeretlen tényezőre van szükség, amelyet csak erős általánosítással lehetne körülírni. A probléma megjelenésekor kezdtem kutatni olyan alternatív megoldások felé, ahol szimulációs megoldásokkal lehetővé válhat a valós helyzetek lemodellezése és meghatározhatók az adott körülményhez tartozó eredmények. Lehetőség nyílik különböző szituációkat meghatározni, és felépített környezetben a valóságnak megfelelően lemodellezni. Ennek eredményeképpen a szimulációból kiolvashatók lehetnek például az adott pályaszakaszra és az adott körülményekre (mint például érvényben lévő lassúmenetek jellege és száma) vonatkozó energiafogyasztási és visszatáplálási értékek a különböző vezetési stílusokhoz. Az eredmények kiértékelésénél az adódó gyorsítási erőket összehasonlítom egy valós, közelítő mérés eredményével.

Szeretném megköszönni Dr. Horváth Richárd tanár úrnak, hogy elfogadta a felkérésemet a belső konzulensi szerepre, valamint, hogy tudásával és szakértelmével folyamatosan támogatott a kutatásom során. Szeretném továbbá megköszönni a Rail Cargo Hungaria Zrt. Vontatás szervezétének, hogy a szakértelmükkel folyamatosan hozzájárultak információkkal vagy éppen ötletekkel a dolgozatom megvalósulásához.

1. A szimulációk

A szimuláció olyan vizsgálat, amikor egy rendszer, folyamat fizikai vagy számítógépes modelljén tanulmányozzuk a rendszer várható, illetve valódi viselkedését. Az egyik értelmezés alapján tehát a szimuláció jelentése, hogy egy alternatív környezetet létrehozva képesek vagyunk a valóságot leképezni. Ebből adódik, hogy képessé válunk a való életben szükséges vizsgálatokat ebben a fiktív környezetben elvégezni a valós környezet zavarása nélkül. Megváltoztathatunk bármennyi paramétert, amely hatásait vizsgálva hozhatunk döntéseket.

Kezdetben a szimulációs környezeteket egyedileg építették fel, minden problémára személyre szabott megoldást kellett megalkotni. Később már számos kész, kifejezetten közlekedési szimulációs program áll rendelkezésünkre, hogyha egy adott keresztmetszetet szeretnénk felépíteni és megvizsgálni. Néhány példát említve lehetőségünk van a város növekedését figyelembe véve modelleket alkotni, és modellezni a várható forgalmat. Eszköztárunkban szerepelnek a lámpás kereszteződések, és az eltérő fázisrendszerek átbocsájtó képességének vizsgálata. Elérhetők számunkra kifejezetten vasúti közlekedés modellezésére fejlesztett szimulációk, ahol elsősorban üzemviteli számításokat vizsgálhatunk meg, mint például egy vonal átbocsájtó képessége, állomási vágány foglaltsága vagy eltérő vontató járművekkel közlekedő vonatok menetideje.

Ezeknek a kész szimulációs programoknak egy közös általános jellemzője, hogy kívülről nem igazán, vagy csak nagyon körülményes módon lehet manipulálni a modellezési folyamatot. Manipulálás alatt az olyan utasítások adását értem, amelyek összefüggésben vannak a szimuláció aktuális állapotával, vagy az egyes járművek adataival. Sok esetben ezekre a külső behatások érvényesítésére nincs is szükség, hiszen például egy városrész forgalmának általános vizsgálatához nem fontos az egyes járművek vezetőinek pontos döntésének a meghatározása. Abban az esetben, amikor viszont kifejezetten a járművezetői attitűd okozta eltéréseket szeretnénk vizsgálni, elengedhetetlen, hogy külső módszerrel befolyásolhatóvá váljon a szimuláció befolyásolása megadott feltételek alapján.

1.1 A nyílt forráskódú programok irányzata

A nyílt forráskódú rendszerek alatt az olyan rendszereket értjük, amikor az elkészült programcsomagot a felhasználó nem egy zárt környezetként használja, hanem lehetősége van saját igényei szerint tovább fejleszteni, vagy specifikálni. Ezeket a módosításokat a programkód elérhetősége teszi lehetővé. Ezek a rendszerek különleges, szabad felhasználásra vonatkozó licencekkel rendelkeznek.

Ezen rendszerek előnye a sokoldalúságban és a szabadságban nyilvánul meg, ugyanis a felhasználó lehetőséget kap a rendszer teljes mértékben saját igényeinek formálására. Mérnöki számítások során a szimulációs környezetek általában zárt programcsomagként érkeznek, amely miatt az egyedek viselkedéséért felelős algoritmust csak részben értjük meg, és jobb esetben is csak korlátozottan módosíthatjuk.

A szimuláció elkészítése során egy olyan közlekedési szimulációs programmal foglalkoztam, amely nyílt forráskódú, így lehetőségem nyílt a jelen dolgozatban megfogalmazott elveknek megfelelően egyedi funkciókat és eljárásokat kialakítani.

1.2 Közlekedési szimulációs rendszerek

A közlekedési szimulációs rendszerek kifejezetten közlekedési és mobilitási mérések elvégzésére jöttek létre. A modell létrehozásánál lehatárolunk egy adott részletet a valós környezetből, a számunkra éppen fontos információkkal. A rendszerek alkalmasak ezen a leképzett modellen a valóságban előforduló eseményeket utánozni, amely kimeneteit folyamatosan képesek vagyunk elemezni és következtetéseket levonni belőlük.

Megkülönböztethetünk makroszkopikus, mezoszkopikus és mikroszkopikus modelleket, ahol:

[1]

- Makroszkopikus: a forgalom egy közeg ellenállásaként van kezelve; Kiválóan alkalmas hálózati szintű vizsgálatra.
- Mezoszkopikus: kapcsolatot teremt a makro- és a mikroszkopikus modell között; Leginkább dinamikus forgalomszétosztás vizsgálatára alkalmasak.
- Mikroszkopikus: minden jármű külön egyedként szerepel; Jellemzően a jármű-járművezető egy modellként jelenik meg benne.

A dolgozatomban mikroszkopikus modellt alakítottam ki, ahol a mérés mindig egy adott jármű-járművezető modellre vonatkozik.

1.2.1 PTV Vissim

A PTV szoftvercég egyik szimulációs programja a Vissim. A PTV hivatalos weboldala alapján multimodális közlekedés szimulálására alkalmas, ezáltal lehetőség van közlekedési alágazatoktól függetlenül modelleket létrehozni. Képes a városi közlekedés modellezésére, ahol az egyéni közlekedés (gyalogosok, kerékpárosok, személygépjárművek), és a közösségi közlekedés (autóbuszok, trolibuszok, villamosok, városi gyorsvasutak, vasutak) jelennek meg.

A Vissim mikro- és mezoszkopikus modell létrehozására alkalmas. Mikroszkopikus modellezés esetén a legmagasabb a részletezettség, a modell minden gyalogost és járművet egyedként kezel. Lehetőség nyílik a járművek mozgásának bizonyos paramétereinek megváltoztatására (mint például az engedélyezett sebesség). A szimuláció képes az egyes járművek és gyalogosok adatait kimenteni (mint például fogyasztási adatok). Egy konkrét esetet kiragadva, méréseket lehet végezni a forgalomirányító jelzőlámpák fázisainak hatásaival kapcsolatban. Optimalizálni lehet többek között a lámpafázisokat, és megvizsgálhatjuk a torlódások és az utazási idők megváltozását a fázisok változásának függvényében. Ebben a környezetben a szimuláció sokkal nagyobb erőforrást igényel az alacsonyabb részletezettségi módoknál. [2]

Mezoszkopikus modell esetén a részletezettség már alacsonyabb, kevesebb erőforrást igényel így a szimuláció felgyorsítható.

Mindkét típusú szimuláció futása során a járművek mozgását és döntéseit a Vissim befolyásolja. Lehetőség van beállítani számtalan paramétert, ilyen például a járművek vonóerőgörbéjének kézi bevitele, és maximális, illetve kívánt gyorsulás megadása, azonban ezeknek az alkalmazásáról a futtatás során a Vissim dönt. Ugyan COM interfészen keresztül lehetőség van a döntések felülvezérlésére, azonban ez bonyolult és összetett folyamatokat igényel.

1.2.2 PTV Visum

A PTV szoftvercég másik programja a Visum, ami egy komplex szimulációs környezet. A PTV hivatalos weboldala alapján a program képes többek között szállítási modelleket, multimodális közlekedési modelleket és közösségi közlekedést szimuláló modelleket létrehozni és elemezni. A Visum makroszkopikus környezet, a szimuláció a hálózati elemeket vizsgálja.

A járművek már mennyiségi, mintsem minőségi adatként jelennek meg. A program alkalmas például területek (városok, kerületek, városrészek, zónák) közötti forgalomnagyság és forgalomáramlás meghatározására, utazási idők és utazási igények megállapítására és a keletkező üzemeltetési költségek kiszámítására.

Mivel a szimuláció során a járművek részletes adatai helyett csak azok számosságát vizsgálja, sokkal nagyobb területek lehatárolására ad lehetőséget. A modellt kialakíthatjuk áruáramlatok vizsgálatára, gyalogosok áramlásának vizsgálatára egy repülőtérre vagy vasútállomáson, logisztikai áramlatok megfigyelésére. [3]

A Visum alkalmas vasúti szimuláció létrehozására is, azonban ez a makroszkopikus jellegéből adódóan áramlatszerű modellezést eredményez. Hasznos lehet vasúti kocsáramlatok meghatározásánál, vonatok tervezésénél és flottakezelésnél.

Mindazonáltal, hogy a Visum jellemzően forgalomnagyságot és forgalomáramlást modellez, általánosságban inkább hálózati mérések szimulálására használható (mint például egy megváltozott célforgalom hatásai egy adott keresztmetszeten). A saját algoritmusait és szimulációra ható utasításait felülvezérelni bizonyos mértékig COM interfészen keresztül lehetséges.

1.2.3 OpenTrack

Az OpenTrack a hivatalos weboldala alapján egy az 1990-es évek közepén kezdődő kutatási projekt eredménye, amelyet a Svájci Szövetségi Technológiai Intézet (Swiss Federal Institute of Technology) indított. A projekt célja volt, hogy objektum-orientált modellezés módján fejlesszen egy olyan szimulációt, amely az összetett vasúti problémákra gyakorlati gazdasági megoldásokat nyújt. Mára az OpenTrack szimuláció egy jól bevált eszközzé vált a vasútállomások, vasúti beszállító cégek, szakértők és az egyetemek körében.

A szimulációban többek között lehetőség nyílik modellezni, elemezni és szimulálni a: nagysebességű vasutat, normál vasutat (személy és teherszállítás egyaránt), bányavasúti rendszereket, metrókat, LRT (Light Rail) rendszereket. A szimuláció a következő paraméterek kiszámítására és vizsgálatára kínál lehetőséget: [4]

- Az infrastruktúra követelményeinek meghatározása,
- A vasútállomások és vasútvonalak kapacitásának meghatározása,
- Minimális fázisidő és követési idő kiszámítása,

- Gördülőállományra vonatkozó számítások,
- Menetrend készítése és azok elemzése a szimuláció során,
- Különböző biztosítóberendezési rendszerek szimulálása (állomástávolság, önműködő térköz, mozgótérköz, ETCS L1-L3 és egyéb további rendszerek)
- Biztosítóberendezési hibák és egyéb zavartató tényezők hatásainak elemzése,
- A vontatás teljesítmény- és energiaigényének meghatározása,
- Felsővezetéki rendszer betáplálásának szimulálása.

A dolgozatom témájába illeszkedve a korábban felsorolt funkciók közül most a vontatás teljesítmény- és energiaigényének meghatározását emelem ki. Az OpenTrack lehetőséget ad különböző mozdonytípusok meghatározására, így minden mozdony karakterisztikája (vonóerőgörbe, villamosfékérő görbe, tömeg, hosszúság, adhéziós tényező és vontatási nem) eltárolható. A leírtaknak megfelelően tehát ezen a ponton a modellezés során lehetőségünk van a mozdony paramétereit a valóságnak megfelelően beállítani. Ez a beállítás azonban egy vizuális felületen történik, ahol a kurzor segítségével alakíthatjuk a kirajzolt vonóerő- és villamosfékérő görbét. A látszólagosan beállított értékek a valóshoz csak közelítenek.

Az OpenTrack szimuláció biztosít API (vagy más néven: Application Programming Interface, továbbiakban: interfész vagy API) lehetőséget, amellyel lekérdezhetünk például vonathoz tartozó paramétereket, és befolyásolhatjuk például egy vonat kívánt sebességét. Ez a manipulálási lehetőség a jelen dolgozatban foglalt témát illetően nem elegendő, hiszen nincsen lehetőség például az éppen rendelkezésre álló vonóerőt felülvezérelni. Ugyanakkor nem találtam arra vonatkozóan információt, hogy a szimuláció milyen feltételek szerint dönt a vonóerő és villamosfékérő kivezérlésekor annak mértékéről.

1.2.4 Simulation of Urban Mobility

A hivatalos weboldal információi alapján a Simulation of Urban Mobility (vagy más néven: SUMO) az Eclipse cég ingyenes és nyílt forráskódú, operációs rendszerektől független forgalomszimulációs programcsomagja. A szimulációs környezet 2001 óta érhető el, és lehetőséget ad intermodális hálózatok és közlekedési alágazatok modellezésére. A SUMO az alapértelmezett lehetőségeken túl bővíthető egyéni modellekkel és számtalan API lehetőséget biztosít számunkra. [5] Az API olyan külső kapcsolatot tesz lehetővé egy futó programkörnyezetben, ami előre meghatározott módokon megengedi a program futásának

kívülről történő befolyásolását. A manipulációs folyamat lehet lekérdezés, módosítás vagy egy bizonyos funkció meghívása.

A SUMO alkalmazási területe a programcsomag ingyenességéből és szabadon tovább fejleszthetőségéből adódóan nagyon szerteágazó, nagy volumenű projektek megvalósítására és optimalizálásra alkalmas. A programcsomag néhány, a jelen dolgozat tematikájába illeszkedő felhasználási lehetőség: [5]

- Mesterséges Intelligencia tanítása közlekedési lámpaprogramok tervezésére,
- Vasúti forgalom szimulálása a mesterséges intelligencia által irányított vasúti pályán,
- Forgalom biztonság és veszélyeztetések elemzése,
- Környezetszennyezési számítások (zajszennyezés és levegőszennyezés).

1.2.4.1 SUMO alapkörnyezetei

A SUMO szimuláció futtatására lehetőségünk van parancssorból vagy grafikus felületen. Előbbi esetében a szimuláció csak és kizárólag háttérfolyamatként fut le, és a végeredmény a konfigurációs beállításoknak megfelelően kerül kimentésre. A grafikus futtatás esetében lehetőségünk van vizuálisan is nyomon követni a szimuláció állapotát. Megállíthatjuk, gyorsíthatjuk vagy lassíthatjuk a szimulációt, skálázhatjuk a hálózaton lévő járművek számát. Ebben a módban lehetőségünk van az egyes járműveket valós időben megfigyelni, és különféle grafikon nézeteket állíthatunk elő a dinamikus értékeikből. [6]

A szimuláció elindításához elengedhetetlenül szükséges létrehozni a hálózati topológiát (vasúti pálya, váltók, jelzők), valamint a forgalmi igényeket (járművek, útvonalak, járatok). Ezek az elemek egy *xml* fájlban történnek rögzítésre, melyek létrehozása történhet a *netedit* grafikus szerkesztővel, vagy manuálisan begépelve. Végeredményében a SUMO futtatásához erre a két fájlra elengedhetetlenül szükség van. [6] A SUMO-ban lehetőségünk van egy központi konfigurációs fájl (SUMOconfig) elmentésére, amely önmagában tartalmazza az összes szükséges *xml* fájl elérési útvonalát, ezáltal a szimuláció egy fájlból is betölthetővé válik.

Az *xml* fájlokban eltárolt elemek statikusak, így a szimuláció futtatása során alapkörnyezetben a paraméterek nem módosíthatók. A járművek a statikusan rögzített paramétereknek megfelelően fognak döntéseket hozni és közlekedni. Az 1.2.4.2 fejezetben bemutatom a TraCI-t, amellyel lehetőség nyílik e statikus paraméterek online módosítására.

1.2.4.2 A SUMO működését kiegészítő TraCI elem bemutatása

A Traffic Control Interface (vagy más néven: TraCI), magyarul Forgalomszabályozó Interfész hozzáférést ad a futó szimulációhoz. Ezzel a hozzáféréssel lehetőség adódik szimulált objektumok értékeinek lekérdezésére, és azok viselkedésének befolyásolására. Az interfész programozási nyelve lehet C++, .NET, Java, Matlab vagy Python is.

Jelen dolgozatban a Python programozási nyelvvel alakítottam ki az interfészt, amelyhez szükségem volt a Python TraCI csomagra. [7]

A TraCI számos paraméter lekérdezését és befolyásolását engedi meg, úgy mint:

- forgalmi objektumok,
- érzékelők és kimenetek,
- hálózat,
- infrastruktúra,
- szimulációs környezet.

A dokumentációban elérhetők az interfészhez kapcsolódó parancsok és utasítások. A jelen munkám során a forgalmi objektumokon belül a jármű értékeinek lekérdezésével, megváltoztatásával kapcsolatos parancsokat használtam. Ezeket a főbb parancsokat az 1.1. táblázat tartalmazza.

A munkám során egy meghatározott vontatójármű mozgását vizsgálom, amelyhez a TraCI interfész adta lehetőségeket veszem igénybe. Meghatározhatóvá válik a pillanatnyilag rendelkezésre álló legnagyobb vonó- vagy villamosfőkerő, amelynek végrehajtására további feltételrendszert állíthatunk fel. Bevezethetünk olyan arányszámokat, amellyel leírhatjuk a valóságban menetszabályzóként és villamosfőkerő szabályzóként ismert mozdonyvezetői kezelőszervek működését a modellünkben. Ezek megvalósításáról az 5.7 fejezetben térek ki részletesen.

TraCI parancs	Funkció
setSpeed	A jármű felgyorsítása egy bizonyos sebességre. [m/s]
getSpeed	A jármű sebességének lekérdezése. [m/s]
setAccel	A jármű gyorsulásának megadása. A jármű ezzel a megadott gyorsulással fogja elérni a beállított vagy megengedhető legnagyobb sebességet. [m/s ²]
setDecel	A jármű lassulásának megadása. A jármű ezzel a megadott lassulással fogja elérni a megadott vagy megengedhető legnagyobb sebességet. [m/s ²]
getIDList	A járművek azonosítóinak lekérdezése.
simulationStep	A szimuláció léptetése.

1.1. táblázat: A TraCI releváns parancsai [7]

2. A vontatójármű és jellemző paraméterei a szimuláció szempontjából

A vasúti vontatójármű (vagy más néven: mozdony) beépített erőforrással rendelkezik, és képes önerőből mozgást végezni. A beépített erőforrás szerint megkülönböztethetünk:

- villamosvontatást,
- dízelvontatást,
- gőzvontatást,
- hibrid vontatást.

Munkám során villamosvontatású vontatójárművel végeztem a számításokat. A vizsgálatok elvégzéséhez a Siemens ES64U2 típusjelölésű, Taurus fantázianevű, vasútállalatoknál azonosítható nevén ÖBB 1116 vagy MÁV 1047 mozdonyt választottam.



2.1. ábra: Az ÖBB 1116-os Taurus mozdonya (Forrás: saját fénykép)

A vasúti vontatott járművek (vagy más néven: vasúti kocsik; összességét tekintve: vontatmány vagy elegy) nem rendelkeznek önálló erőforrással, továbbítását a vontatójármű végzi. Jelenlegi szimuláció során a vontatott járműállomány adatai közül csak és kizárólag az eredő elegytömeget vettem figyelembe, ezen szimuláció során nincsen még jelentősége a vontatmány homogén vagy heterogén jellegének. A dolgozat során többször hivatkozom a szerelvény kifejezésre, ami a vontatójármű és vontatott járművek egységét jelenti.

2.1 A vontatójármű alaptulajdonságai

A választott mozdony villamosvontatású, váltakozó áramnemű rendszerben, kétfrekvencián képes közlekedni, amelyek:

- 25 kV 50 Hz (például a magyarországi hálózat is),
- 15 kV 16 2/3 Hz (többek között a németországi és az osztrák hálózat).

A jármű korszerűnek tekinthető, képes a visszatáplálásra és úgynevezett AFB üzemmóddal rendelkezik. Az AFB üzemmód bekapcsolásakor a sebesség alapjel-szabályzóval beállított sebességre a mozdony a vonóerő automatikus szabályozásával felgyorsít, lelassít, sebességét tartja a vonóerő vagy fékezőerő szabályozásával az adott körülményeknek megfelelően. A menetszabályzó kar állásával szabályozható a maximálisan kivezérelhető vonóerő, amelyet a mozdony AFB üzemmódban is figyelembe vesz. [8]

Dolgozatom során a szimuláció futtatásánál a kívánt sebesség elérését úgy modellezem le, mintha a mozdonyvezető az AFB használatával jelölne ki egy kívánt sebességet, és a menetszabályzó kart egy statikus pozícióban tartaná. Ebből kiindulva tehát a szimuláció során a mozdony érzékeli a pillanatnyi és kívánt sebességet, a vonóerőt pedig a sebességnek, a menetszabályzó kar fiktív állásának és a pillanatnyilag elérhető maximális vonóerő alapján határozza meg.

2.1.1 A választott jármű jellemző teljesítmény adatai

A Siemens ES64U2 mozdony kezelési útmutatója alapján a jármű releváns főbb műszaki jellemzőit a 2.1. táblázat tartalmazza.

Konstruktív sebesség	230 km/h
Legnagyobb indító vonóerő	300 kN
Állandó vonóerő	250 kN
Maximális villamosfék erő	150 kN
Maximális teljesítmény („Tartalékkal”)	7,0 MW
Állandó teljesítmény	6,4 MW
Mozdony tömege	86 vagy 88 tonna, kiépítéstől függően
Ütközők közötti hosszúság	19.280 mm
Villamosfék	Felsővezetékbe visszatápláló

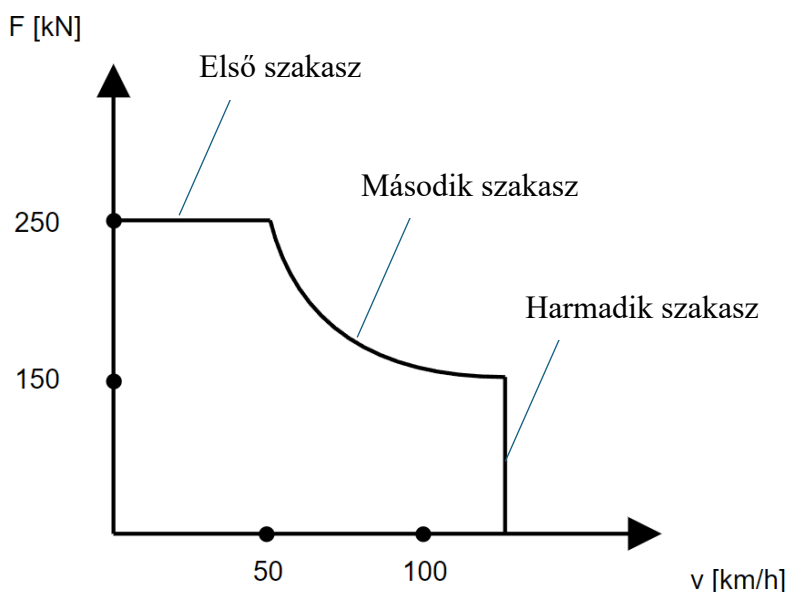
2.1. táblázat: Siemens ES64U2 mozdony főbb műszaki adatai [8]

2.1.2 Vonóerőgörbe

„A vasúti vontatójárművek vonóereje elsősorban a vonóhorgon kifejtett vonóerőben nyilvánul meg, amit méréssel meg tudunk határozni.” [9] Az így létrejött pontszerű adathalmazra folytonos görbét illesztnek, ezzel előállítva a vonóerőgörbét.

A vonóerőgörbe egy olyan diagram, amely vízszintes tengelye a sebesség értékek (v ; km/h), a függőleges tengelye a vonóerő értékek (F ; kN) (lásd: 2.2. ábra).

A 2.2. ábrán egy fiktív vontatójármű vonóerőgörbéje látható. Ez alapján a görbét általánosságban három szakaszra tagolhatjuk. Az első szakaszban, a példában konstans 250 kN értéket határoztunk meg a mozdony maximális vonóerejének. Ez az érték lehet szoftveres korlátozás is vagy az adhéziós vonóerő túllépése miatti lehatárolás. [10] A második szakaszban, 50 km/h óra felett egy görbe alkotja a vonóerőgörbét, amelyre az adatokat elemeire bontva trendvonalat illeszthetünk, mely képletét felhasználva meghatározhatjuk azt a függvényt, ami a görbét kirajzolja. Dolgozatom során ezzel a módszerrel készítettem az eredetihez közelítő vonóerőgörbét a választott járműhöz. A fiktív görbe körülbelül 130 km/h óra sebességig alkotja a második szakaszt, majd e sebesség felett egy konstans 0 kN értéket vesz fel a vonóerő. Ez a hirtelen beesés a mozdony konstrukciós sebességéből adódik, a vonóerőgörbét jellemzően a mozdony tervezett legnagyobb sebességéig határozzák meg, mivel e sebesség felett nincsen üzemi jelentőségük.



2.2. ábra: Egy fiktív jármű vonóerőgörbéje (Forrás: saját készítés)

A vonóerőgörbék általános jellemzője, hogy az első szakaszban egy konstans értéken, vagy egy lineáris görbe mentén változik a maximális vonóerő. Ezt követi egy exponenciális szakasz, amely képletét értékpárok alkotásával, és trendvonal illesztésével határoztunk meg. A harmadik szakasz egy konstans 0 érték, ugyanis a konstrukciós sebességnél nagyobb sebességeknél nem értelmezik a mozdony maximális vonóerejét.

2.1.2.1 A választott mozdony vonóerőgörbéje

Az általam alapértelmezettként meghatározott vontatójármű vonó- és villamosfőkerő görbéjét a 2.3. ábra szemlélteti.

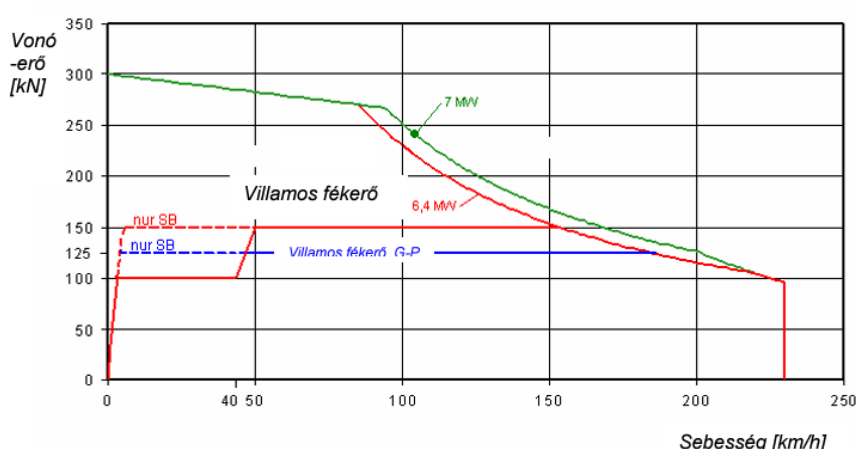


Abb. 3-1: Z/V-Jelleggörbe
B/V-Jelleggörbe

2.3. ábra: Az ES64U2 mozdony vonó- és villamosfőkerő görbéje [8]

A 2.3. ábra tartalmazza egyaránt a vonóerőgörbét, valamint a villamosfőkerő görbét. A zöld színnel jelölt 7 MW teljesítmény a következő üzemmódban érvényesül: „A képernyő „Tartalék” nyomógombjának megnyomása esetén a hajtásvezérlés 5 perc időtartamban a maximális teljesítményt 6,4 MW-ról 7 MW-ra növeli. Mint ahogy a vonóerő-sebesség jelleggörbéből is látható a teljesítmény növekedés csak 85 km/h-tól 200 km/h-ig ad le nagyobb vonóerőt.” [8]

A 0 km/h és 85 km/h közötti zöld szakasz magába foglalja a normál üzemmódot, amely a későbbiekben piros színnel van ábrázolva. A szintén piros színnel ábrázolt, 50 km/h és körülbelül 150 km/h között konstans 150 kN értéket mutató görbe a villamosfőkerő görbe. Jól kivehető, hogy a villamosfőkerő nem képes önmagában teljesen megállítani a járművet hiszen 5 km/h és 0 km/h között lineárisan 100 kN-ról 0 kN-ra csökken. Látható továbbá, hogy 50 km/h és 40 km/h óra között a villamosfőkerő görbe szintén lineárisan csökken.

Körülbelül 5 km/h és 40 km/h között konstans 100 kN értéken van, kivéve „nur SB”, vagy másnéven gyorsfékezés esetén. Ekkor 15 km/h óráig 150 kN villamosfékerőre képes a mozdony, majd ezt követően kezd az erő csökkenni. A kék színű görbe G-P féküzemmódban való közlekedéskor érvényes, ilyenkor 150 kN helyett 125 kN a maximális villamosfékerő.

A vonóerőgörbe leképzésénél eltekintettem a „Tartalék” opciótól, valamint figyelmen kívül hagytam a gyorsfékezés esetén, és a G-P féküzemmód esetén érvényes értékeket.

3. A vonóerő- és villamosfékerő görbe leképezése

A görbe függvénnyé alakításánál első lépésként az elérhető vonóerő- és villamosfékerő görbe alapján a vonóerőgörbét (lásd: 2.3. ábra) három részre bontottam. Erre a 2.1.2 fejezetben részletezett okoknál fogva van lehetőségünk.

3.1 A vonóerőgörbe leképezése

A vonóerőgörbe három része a következő szakaszokból áll:

1. 0 km/h és 85 km/h közötti szakasz,
2. 85 km/h és 230 km/h közötti szakasz,
3. 230 km/h feletti szakasz.

A 3.1 fejezet alpontjaiban a fentebb megnevezett szakaszok leképezését ismertetem részletesen.

3.1.1 Az első szakasz leképezése

Az első szakaszt 0 km/h és 85 km/h között értelmezzük. Ebben a szakaszban egy lineáris csökkenés van, 300 kN indító vonóerőről körülbelül 270 kN-ig. Ezen lineáris függvény értékét az (1) képlettel számíthatjuk ki, amely visszaadja a mozdony maximális elméleti vonóerejét ebben a sebességtartományban.

$$\text{Ha } 0 \leq v < 85 \qquad F = 300 - \frac{v}{2,833333} \qquad (1)$$

ahol:

- F : a vonóerő kN-ban,
- v : a jármű sebessége km/h-ban.

3.1.2 A második szakasz leképzése

A második szakaszt a 85 km/h és 230 km/h sebességi értékek között értelmezzük. Ezen szakasz leképzése összetettebb feladat, amely során a trendvonal illesztését programozási megoldásokkal végeztem el.

Első lépésként 85 km/h-ról indulva 230 km/h-ig 10 egységnyi szakaszokat vettem fel a vízszintes sebesség tengely mentén, és ugyanezen elv alapján 5-10 egységnyi szakaszokat vettem fel a függőleges vonóerő tengely mentén is. Ezeknek a szakaszolásoknak köszönhetően hozzávetőlegesen kialakítottam az értékpárokat, minden sebességi értékhez hozzárendeltem egy a fent említett módszerrel megállapított vonóerő értéket (lásd: 3.1. táblázat).

v [km/h]	85	93	100	110	120	130	140	153	160	170	180	190	200	210	220	230	<
F [kN]	270	250	230	210	190	180	170	160	150	145	130	125	120	115	107	97	0

3.1. táblázat: A mozdony vonóerőgörbéjéből meghatározott értékpárok (Forrás: saját számítás)

Az értékpárok létrehozásának oka, hogy azokra egy trendvonal illesztéssel megállapíthassuk a függvényt, amely reprezentálja a vonóerőgörbét ebben a szakaszban. A trendvonalat a *SciPy* és *NumPy* Python csomagok használatával számítottam ki, amelyet a 3.1.2.1 fejezetben fejtem ki.

3.1.2.1 Trendvonal illesztése Python csomagokkal

A *SciPy* és *NumPy* csomagokkal való számítást a *Jupyter Notebook* interaktív programozási környezetben végeztem el, Python programozási nyelvben. A 3.1. táblázat értékpárjait két változóban tároltam el, amelyekre a *SciPy* az illesztőfüggvényt közelítette, és végül meghatározta a legpontosabb paramétereket. Az illesztőfüggvényt futtató és pontosító programkód megírásában a *SciPy* és a *NumPy* felhasználói útmutatóin kívül segítségként használtam a ChatGPT 3.5-ös verzióját is.

Az exponenciális formulát, amely illeszthető az értékpárokra a (2) egyenlet szemlélteti:

$$f(x) = a * e^{b*x} + c \quad (2)$$

ahol:

- a , b és c a paraméterek,
- x a változó.

Az illesztőfüggvény számítása során a következő paraméterek közelítették a legjobban az értékpárokat:

- $a = 561,5054293$
- $b = -0.01252106$
- $c = 72.21850167$

A függvénybe behelyettesítve a 85 km/h és 230 km/h közötti szakaszra a (3) képlettel határozhatjuk meg a maximális rendelkezésre álló vonóerőt. A paramétereket két tizedesjegyre kerekítve szemléltetem, azonban a szimulációs folyamatokban a pontos eredmény van figyelembe véve.

$$\text{Ha } 85 \leq v \leq 230 \quad F = 561,51 * e^{-0,013*v} + 72,22 \quad (3)$$

ahol:

- F : a vonóerő kN-ban,
- v : a sebesség km/h-ban.

3.1.3 A harmadik szakasz leképzése

A 230 km/h feletti szakaszt leképezni egy konstans 0 kN érték felvételével lehet. Az eddigi gondolatmenetet követve a harmadik szakaszra vonatkozó maximálisan rendelkezésre álló vonóerőt a (4) képlettel lehet meghatározni.

$$\text{Ha } 230 < v \quad F = 0 \quad (4)$$

ahol:

- F : a vonóerő kN-ban.

3.2 A villamosfékerő görbe leképzése

A villamosfékerő görbe (lásd: 2.3. ábra) leképzése közel azonos módszerekkel történik, mint ahogyan a 3.1 fejezetben ismertettem, néhány kisebb eltéréssel. A leképzéseket a 2.1.2.1 fejezetben meghatározott lehatárolásoknak megfelelően ismertetem. A modellezés során ismételten szükségünk van a görbe több részre tagolására, amelyeket a következőképpen határoztam meg:

1. 0 km/h és 5 km/h közötti szakasz,
2. 5 km/h és 40 km/h közötti szakasz,
3. 40 km/h és 50 km/h közötti szakasz,
4. 50 km/h és körülbelül 150 km/h közötti szakasz,

5. körülbelül 150 km/h és 230 km/h közötti szakasz,
6. 230 km/h feletti szakasz.

A 3.2 fejezet alpontjaiban e szakaszok leképzését ismertetem.

3.2.1 Az első szakasz leképzése

Az első szakaszban a maximális villamosfékerő 0 km/h és 5 km/h között 0 kN-ról lineárisan 100 kN-ra növekszik. Ez a villamosfékerő a sebesség függvényében a (5) egyenlettel képezhető le.

$$\text{Ha } 0 < v \leq 5 \qquad F = \frac{v}{0,05} \qquad (5)$$

ahol:

- F : villamosfékerő kN-ban,
- v : sebesség km/h-ban.

3.2.2 A második szakasz leképzése

A második szakaszban 5 km/h és 40 km/h között a maximális villamosfékerő konstans 100 kN.

$$\text{Ha } 5 < v < 40 \qquad F = 100 \qquad (6)$$

ahol:

- F : villamosfékerő kN-ban.

3.2.3 A harmadik szakasz leképzése

A harmadik szakaszban 40 km/h és 50 km/h között a maximális villamosfékerő 100 kN-ról 150 kN-ra növekszik. Ez a villamosfékerő a sebesség függvényében a (7) egyenlettel képezhető le.

$$\text{Ha } 40 \leq v \leq 50 \qquad F = \frac{v}{0,2} - 100 \qquad (7)$$

ahol:

- F : villamosfékerő kN-ban,
- v : sebesség km/h-ban.

3.2.4 A negyedik szakasz leképezése

A negyedik szakaszban 50 km/h és körülbelül 150 km/h között a maximális villamosfékerő konstans 150 kN. Az előző gondolatmenetbe illeszkedve ez a (8) egyenlettel határozható meg.

$$\text{Ha } 50 < v \leq 150 \qquad F = 150 \qquad (8)$$

ahol:

- F : villamosfékerő kN-ban.

3.2.5 Az ötödik szakasz leképezése

Az ötödik szakaszban (körülbelül 150 km/h és 230 km/h között) a maximális villamosfékerő megegyezik az ebben a szakaszban érvényes maximális vonóerő egyenletével. Az ide vonatkozó számítást a (3) egyenlet mutatja, azonban a képlet csak a $(150 < v \leq 230)$ intervallumban érvényes. Ebben az értelmezésben a (3) egyenletben az F tehát maximális villamosfékerő kN-ban.

3.2.6 A hatodik szakasz leképezése

A hatodik szakaszban a maximális villamosfékerő a konstrukciós sebesség feletti tartományban nem ismert, ezért konstans 0 kN értéket veszünk ezen érték felett.

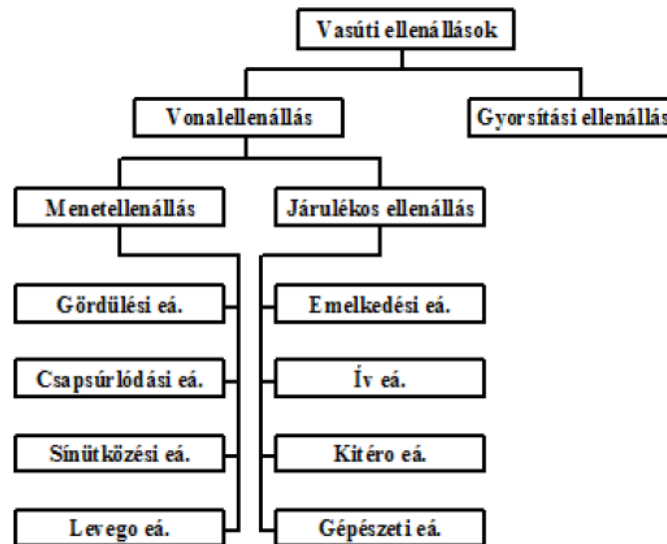
$$\text{Ha } 230 < v \qquad F = 0 \qquad (9)$$

ahol:

- F : villamosfékerő kN-ban.

3.3 Az ellenállások

A vasúti ellenállások (lásd 3.1. ábra) meghatározásánál beszélhetünk vonalellenállásról és gyorsítási ellenállásokról. A vonalellenállásokon belül két nagy csoportot különböztethetünk meg, a menetellenállást és a járulékos ellenállást. A menetellenállás részei a gördülési-, csapsúrlódási-, sínütközési- és a levegő ellenállás. A járulékos ellenállást képező ellenállások az emelkedési-, ív-, kitérő- és gépészeti ellenállás. [11] Ezek meghatározása összetett számításokkal lehetséges, ezért megszokott eljárás, hogy a vasúti üzemeltetők egy csoportonként (például villamosmozdony, dízelmozdony) általánosan igaz függvényt határoznak meg az eredő ellenállás leírására.



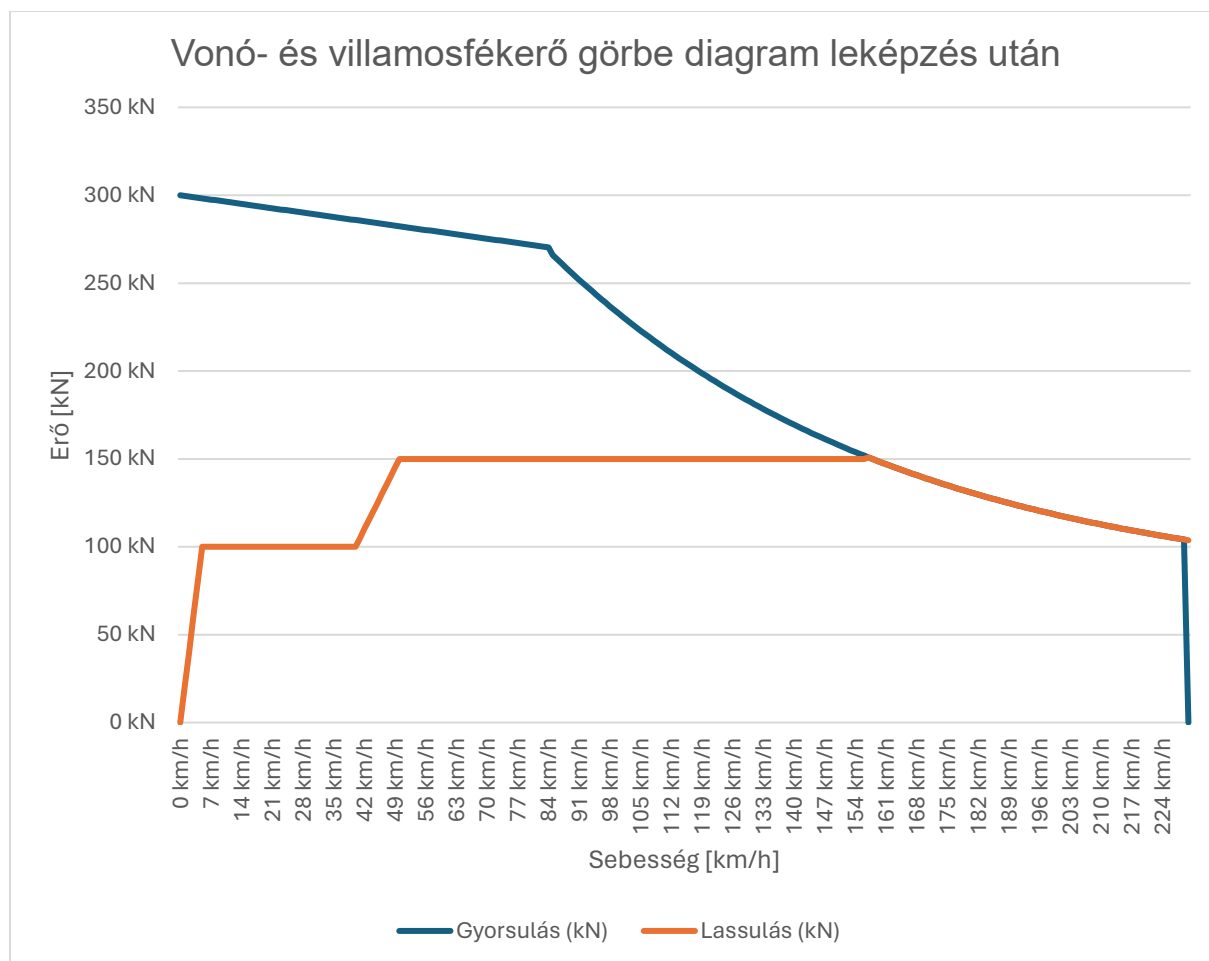
3.1. ábra: Vasúti ellenállások csoportosítása [11]

Jelen dolgozatomban az egyszerűsített járműmozgások vizsgálatával foglalkozom, ezért a munkám során az ellenállásokat az emelkedési ellenállás kivételével nem vettem figyelembe. Mivel a szimulációt sík terepen futtatom, ezért az emelkedési ellenállás nem befolyásolja a mérési eredményeimet, gyakorlati értelemben ennek számítására sem került sor a szimuláció folyamán.

3.4 A vonó- és villamosfékerő görbe felhasználása

A vonó- és villamosfékerő görbe leképezése során minden elválasztott szakaszhoz meghatározásra került egy egyenlet, amely a sebességi értéket behelyettesítve képes visszaadni az adott sebességnél érvényes maximális vonóerőt vagy villamosfékerőt. Fontos szempont, hogy az eredmény akkor lesz valós, hogyha a számozott képletek előtti feltételek teljesülnek. A 3.1 és 3.2 alfejezetekben bemutatott leképzési módszerek eredményének tényleges felhasználását és integrálását a TraCI interfészbe az 5. fejezetben mutatom be.

A leképzésre vonatkozó számításokat egy diagrammon ábrázolva kirajzolódik a vontatójármű vonó- és fékerőgörbe diagramja, amelyet a 3.2. ábra tartalmaz.



3.2. ábra: A mozdony vonó- és villamosfékerő görbéje a leképezés során kiszámított értékekkel ábrázolva (Forrás: saját ábra)

4. A szimulációs környezet felépítése

A szimulációs környezet a Simulation of Urban Mobility (vagy más néven SUMO [5]) nyílt forráskódú rendszerben épül fel, és kerül lefuttatásra. A szimulációs programról bővebben a 1.2.4 fejezetben tettem említést. A környezet felépítéséhez szükségünk van a program meglétére, és egy a projektnek szánt könyvtárra egy tetszőleges helyen a számítógépen. A környezetet jelen dolgozatomban egy egyenes, körülbelül 150 km hosszú vasúti pálya alkotja, mivel a jelen tanulmány készítése során egyszerűsített vonatmozgásokkal végeztem a vizsgálatokat.

4.1 Futtatási környezet létrehozása

A szimulációhoz létrehozunk egy könyvtárat, majd a *netedit* vizuális szerkesztőt nyitjuk meg. Létrehozuk az új hálózatot, majd elmentjük egy SUMOconfig konfigurációs fájlként, hogy a létrehozott elemeket a későbbiekben csoportosítva tudjuk betölteni. A *Network* programfülön belül úgynevezett *edge* létrehozásával és az üres szerkesztői felületre kattintással elhelyezhetjük a megépíteni kívánt útszakaszunk kezdőpontját.

4.1.1 Vasúti pálya felépítése és beállítása

Vasúti pálya létrehozásához a *Network* programfülön belül kiválasztjuk a *Set create edge mode* opciót, majd a *Template selector* szerkesztőnél a *Use edgeType/template* funkciót. Az *Add* opcióval létrehozunk egy újabb típusú utat, majd az *allow* lehetőségre kattintva kiválasztjuk az *Allow only rail vehicles* lehetőséget. Ezután a szerkesztői felületen a kurzor bal gombjának lenyomásával elhelyezzük az út kezdőpontját, majd tetszőleges ponton a végpontját, ezzel létrehozva a vasúti pályánkat. A *speed* mezőben meghatározhatjuk az adott pályaszakasz megengedett legnagyobb sebességét alapértelmezetten m/s-ban. A dolgozatom során 63,88 m/s került meghatározásra, azonban ennek gyakorlati jelentősége ebben az esetben nem lesz.

A SUMO lehetőséget ad a vasúti pálya több paraméterének megváltoztatására is, mint például az egyirányú vagy kétirányú forgalom meghatározása, vagy a pálya legnagyobb megengedett sebességének beállítására. Ezekre a beállításokra ebben a dolgozatban nem térek ki. [12]

4.1.2 Útvonal létrehozása a vasúti pályán

Útvonal létrehozásánál a *Demand* programfülön kiválasztjuk a *Create route mode* opciót, majd a *Route mode* listáknál a második legördülő listából a *rail* lehetőséget. Ennek hatására a megrajzolt vasúti pályánk zöld színnel kitöltésre kerül, amely jelzi az útvonal létrehozásának lehetőségét. Létrehozáshoz a kurzor bal gombjával a pályára kattintunk. A jóváhagyáshoz a *Finish route creation* opciót szükséges választani.

4.1.3 Járműtípus létrehozása a környezetben

Az alapértelmezett járműtípusok között nem szerepel vasúti vontatójármű, ezért azt hozzá kell adnunk a járműtípusokhoz. Ehhez a *Demand* programfülön a *Create type mode* opciót választva a *Create type* funkciót kell használnunk. A létrejövő *t_0* azonosítójú jármű típusát megváltoztatjuk, úgy, hogy a *vClass* mezőben a *rail* kerül kiválasztásra. A járműtípus létrehozása során lehetőségünk van számos paramétert beállítani, mint például a jármű megengedett legnagyobb sebessége. Ezeket az értékeket nem változtattam meg az alapértelmezetről. A dolgozat további fejezeteiben ezen beállítások az interfészen keresztül kerülnek módosításra, így az itteni további beállítási lehetőségeket nem használtam fel.

4.1.4 Jármű hozzáadása az útvonalhoz

Azt, hogy a létrehozott járművünk a létrehozott pályán leközeledjen sokféleképpen elérhetjük a SUMO-ban. Ebben a fejezetben az elérhető sok módszer közül azt az egyik lehetséges megoldást ismertetem, amellyel a szimulációt felépítettem. A *Demand* programfülön a *Create vehicle mode* opcióval, a *Vehicle* legördülő listából a *vehicle (over route)* lehetőséget kiválasztjuk. Ezt követően a *Parent vType* legördülő lehetőségek közül kiválasztjuk a 4.1.3 pontban létrehozott járműtípust (jelen esetben a *t_0* azonosítóval rendelkezőt). Ezután a létrehozott pályaszakaszra kattintva létrejön a jármű a felépített pályán. A jelen szimulációs vizsgálathoz az alapértelmezett értékeket hagytam beállítva, de a *departSpeed* mezőben 0.00 értéket célszerű felvenni. Ez a jármű a hálózatban megjelenéskori sebességét szabályozza.

A jármű hozzáadása után, a konfigurációs fájl megfelelő mentését követően a szimulációs környezet készen áll a szimuláció futtatására.

5. A szimuláció beállítása és specifikálása

A 4. fejezetben bemutatott lépések mentén létrehoztam a szimulációs környezetet. Ennek megfelelően felépítettem az egyszerű pályahálózatot, vasúti járműtípust adtam hozzá a konfigurációs fájlhoz, továbbá a létrehozott útvonalhoz rendeltem a járművet. A szimuláció futtatása lehetséges parancssorból is, vagy grafikus felületen. Akár grafikus, akár parancssoros „egyszerű” futtatással a jármű az alapértelmezetten megadottaknak megfelelően viselkedne, ezért például a gyorsulása konstans $0,1 \text{ m/s}^2$ lenne függetlenül a mozdony sebességétől. A jármű felgyorsulna arra a sebességre, amely a legalacsonyabb a mozdonyál alkalmazható legnagyobb sebesség és a pályasebesség közül. Ennek megváltoztatására online a jelenlegi eszköztárunkban nincsen lehetőség, a szimuláció a szerkesztőben vagy a metaadat fájlokban rögzített paramétereknek megfelelően fog közlekedni.

Ahhoz, hogy a futó szimulációt online, futtatás közben legyen lehetőségünk felülvezérelni TraCI interfészt kell létrehoznunk. Ebben a fejezetben részletesen kitérek az interfész létrehozására, valamint azokra a programozási megoldásokra, amelyekkel megvalósítható a valós jármű adatokon alapuló szimuláció.

5.1 TraCI interfész létrehozása

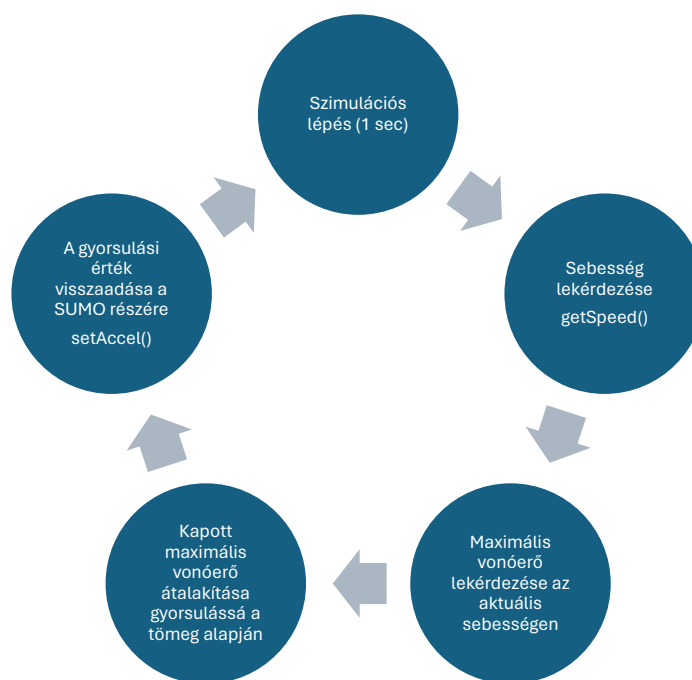
A TraCI interfészről bővebben a 1.2.4.2 fejezetben tettem említést, ahol részleteztem az alaptulajdonságait és funkcióit. Az online interakciókhoz szükségem volt a Python programozási nyelv ismeretére és meglétére a számítógépen. Szükség van továbbá a *traci* python csomagra, amelyet a python fájl létrehozása után importáltam a jövőbeli fő futtatási fájlba. A SUMO és a szimuláció konfigurációs fájljának elérési útvonalának megadása után elindítjuk az interfészt és a SUMO-t a *traci.start()* paranccsal. [13]

Ezzel a lépéssel megnyílik a SUMO grafikus futtatási környezete (vagyis a SUMO-GUI), azonban a TraCI interfész már működik, a python fájlban definiált parancsok végrehajtása a megfelelő körülmények között és a megfelelő időpontban meg fog valósulni az indítást követően. A TraCI interfész aktiváltságát a jobb alsó sarokban megjelenő TraCI felirat és a konfigurációs fájl hiba nélküli betöltése jelzi vissza.

5.2 Vonóerőgörbe Python függvénybe foglalása

Az 3.1 fejezetben említést tettem a vonóerőgörbe matematikai formában történő leképzéséről. Ehhez a vonóerőgörbét három szakaszra bontottam és ezen szakaszokat képeztem le függvény formájában. A három szakasz lebontásánál is az egyes függvények feltételrendszerhez voltak kötve, ezzel előkészítve a függvények integrálását programozási környezetbe.

Ahhoz, hogy a szimuláció folyamán a vontatójárművünk ismerje az aktuális időpillanatban rendelkezésre álló maximális elérhető vonóerőt, egy programozási függvényt (vagy más néven eljárást) kell létrehoznunk. Ez a függvény minden szimulációs lépésben meghívásra kerül, és eredményül visszaadja a függvényt meghívó jármű részére az éppen aktuálisan elérhető maximális vonóerejét a sebessége alapján. Ezt követően a tömeg és az elérhető maximális vonóerő alapján kiszámítja a jármű gyorsulását. Ezt a számított gyorsulási értéket a `setAccel()` paranccsal adjuk vissza a SUMO részére, minek alapján a járművünk ezzel az értékkel fog gyorsulni. Ezt a folyamatot grafikusan az 5.1. ábra mutatja bel.



5.1. ábra: Elérhető gyorsulás megadása a jármű számára (Forrás: saját szerkesztés)

Az eljárás meghívása során paraméterként megadjuk a jármű aktuális sebességét és a szerelvény össztömegét. Mivel a SUMO a sebességet m/s , a gyorsulást m/s^2 mértékegységben értelmezi, ezért a sebesség megadása is m/s -ként történik. A programozási függvény egy belső feltételrendszert (szelekciót) állít fel, miszerint a sebesség melyik görbe szakaszhoz tartozik. A besorolást követően az adott szakasz matematikai formulájába való behelyettesítés után

eredményként visszkapjuk az aktuális sebességhez tartozó maximális rendelkezésre álló vonóerőt. A vonóerő meghatározását követően a (10) egyenlet alapján meghatározhatóvá válik a gyorsulás értéke m/s^2 -ben.

$$a = \frac{F}{m} \quad (10)$$

ahol:

- a : gyorsulás m/s^2 -ben,
- F : vonóerő kN -ban,
- m : a szerelvény össztömege kg -ban.

A programkódban szereplő tényleges megvalósítást az 5.2. ábra mutatja meg. Az ábrán szereplő a , b és c változók azok a paraméterek, amelyeket a függvény illesztés során meghatároztunk. A v mint sebesség egy művelettel át lett alakítva km/h értékre. A $mass$ változó a szerelvény tömegét tartalmazza kg -ban.

```
v = v * 3.6 # v was in m/s
if v < 85:
    force = (300 - v/2.83333333) * 1000
elif v >= 85:
    force = (a * np.exp(b * v) + c) * 1000
else:
    force = 0
return force / mass
```

5.2. ábra: Vonóerő és gyorsulás visszaadásának programkódja (Forrás: saját kód)

5.3 Villamosfékerő görbe Python függvénybe foglalása

A villamosfékerő görbe programozási függvénné alakítási folyamata néhány alapvető eltérést leszámítva közel azonos a vonóerőgörbe korábban ismertetett módszerével. Létrehoztam egy olyan eljárást, amely a bemenő aktuális sebességet egy szelekciót követően besorolja a megfelelő görbe szakaszba, majd behelyettesíti a szükséges képletbe, melyet a 3.2 fejezetben határoztam meg. Az eljárás visszaadja az aktuális sebességnél elérhető maximális villamosfékerőt, azonban ezt az értéket lassulássá kell alakítanunk annak érdekében, hogy a SUMO a szimuláció során képes legyen érvényesíteni. Mivel a lassulás értelmezhető negatív gyorsulásként, ezért a (10) képlet alapján meghatározható a lassulás. A negatív gyorsulás értéke a negatív előjelet indokoltá tenné, azonban a TraCI a *setDecel()* paranccsal állítja be a lassulást, amely pozitív értéket vár.

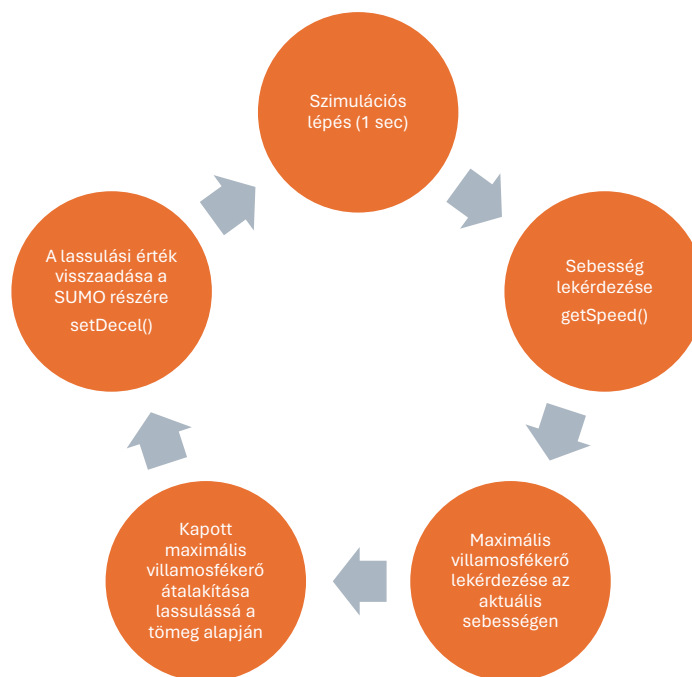
Ezt az értéket a (11) képlet alapján határozhatjuk meg.

$$d = \frac{F}{m} \quad (11)$$

ahol:

- d : lassulás m/s^2 -ben,
- F : villamosfékerő kN-ban,
- m : szerelvény össztömege kg-ban.

Ebben a kontextusban tehát lassulásról, és nem negatív gyorsulásról beszélhetünk. Az elérhető maximális villamosfékerő kiszámítását, átalakítását lassulássá és értékének visszaadását a SUMO részére az 5.3. ábra mutatja be.



5.3. ábra: Elérhető lassulás kiszámítása a jármű számára (Forrás: saját szerkesztés)

A függvény programkódját az 5.4. ábra tartalmazza. A programkódban megtalálható v változó a sebesség km/h -ban, a de_force változó pedig a villamosfékerő (deceleration_force rövidítés alapján). A $mass$ változó a szerelvény tömege kg -ban. A kiszámított erő és a szerelvény össztömege alapján kiszámolt lassulást a TraCI `setDecel()` parancsával adtam vissza a SUMO részére, minden szimulációs lépésben.

```

v = v * 3.6 # v was in m/s

if v <= 5:
    de_force = v/0.05
elif 5 < v and v < 40:
    de_force = 100.00
elif 40 <= v and v <= 50:
    de_force = ((v/0.2) - 100)
elif 50 < v and v < 157:
    de_force = 150.00
elif 157 <= v and v <= 230:
    de_force = a * np.exp(b * v) + c
else:
    return 0.00

de_force = de_force * 1000

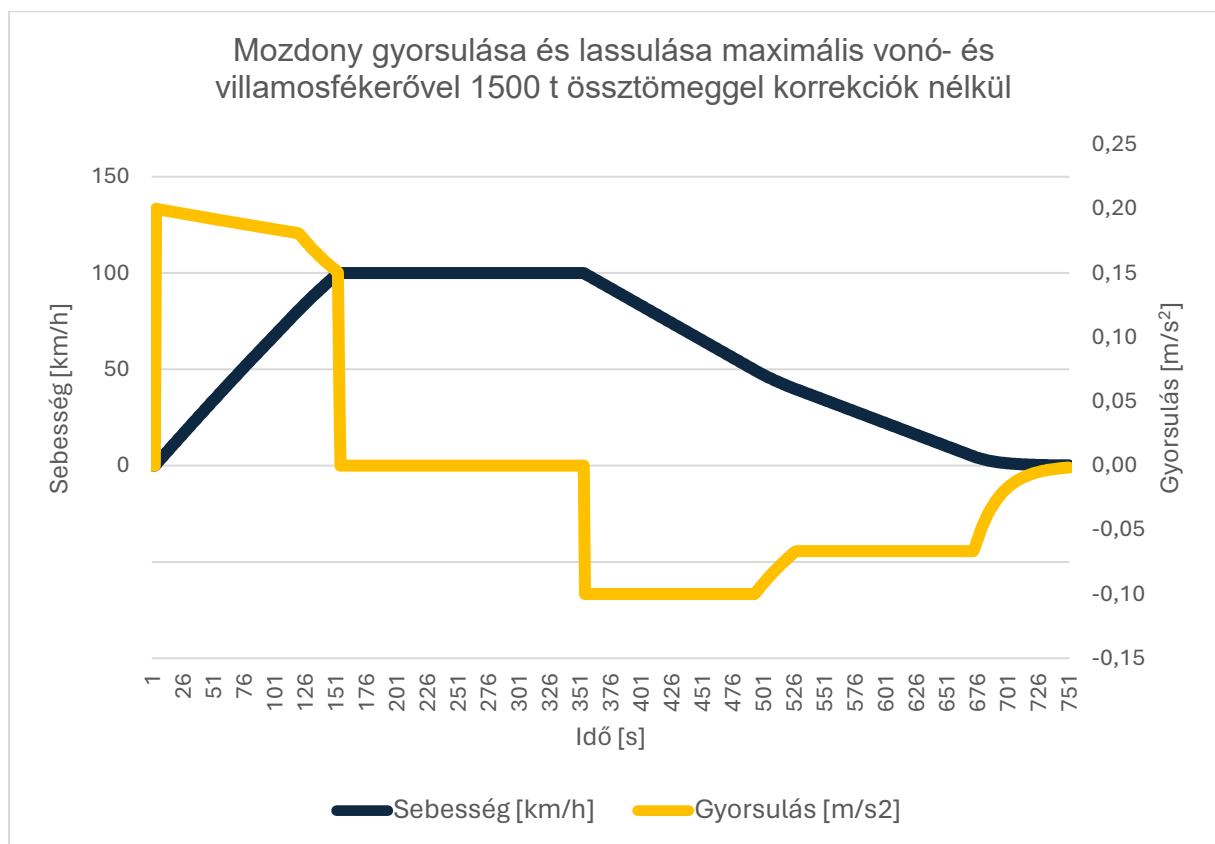
return (de_force / mass)

```

5.4. ábra: Villamosfékerő visszaadása programkódban (Forrás: saját kód)

5.4 A gyorsulás és lassulás finomhangolása

Az 5.2 és 5.3 alfejezetekben szereplő elvek mentén lefuttatott szimuláció eredményét az 5.5. ábra mutatja. Az ábrán a jármű 100 km/h-ra gyorsul, majd kis idejű sebességtartás után szinte 0 km/h-ra lassul kizárólag villamosfékezéssel.

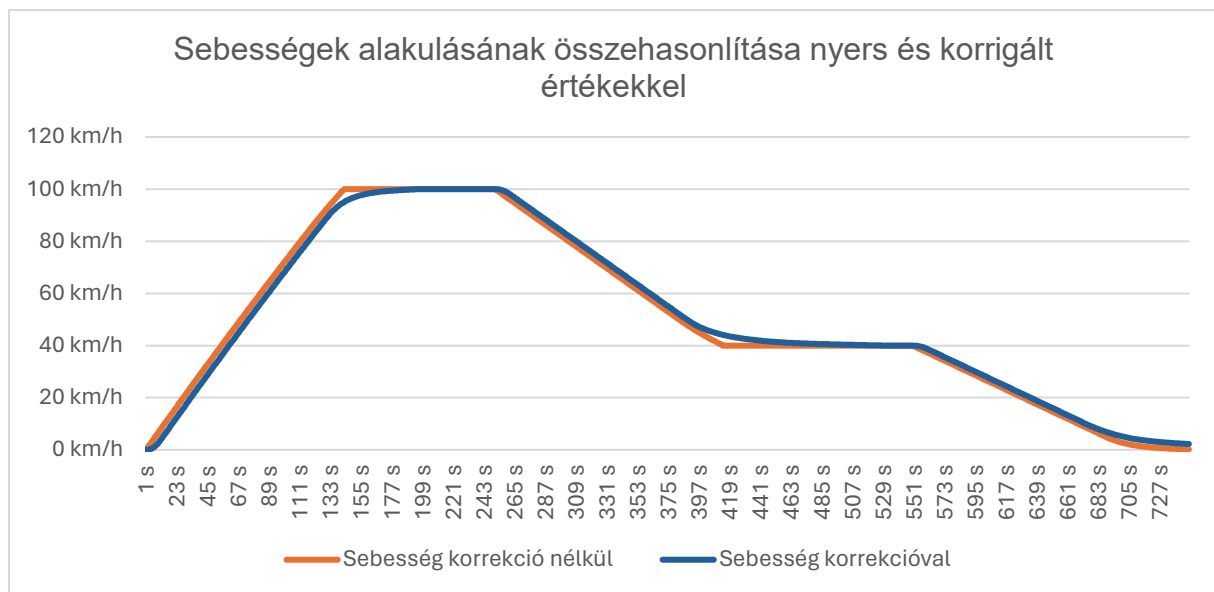


5.5. ábra: Mozdony gyorsulása és lassulása maximális vonóerővel és villamos fékerővel korrekció nélkül (Forrás: saját szerkesztés)

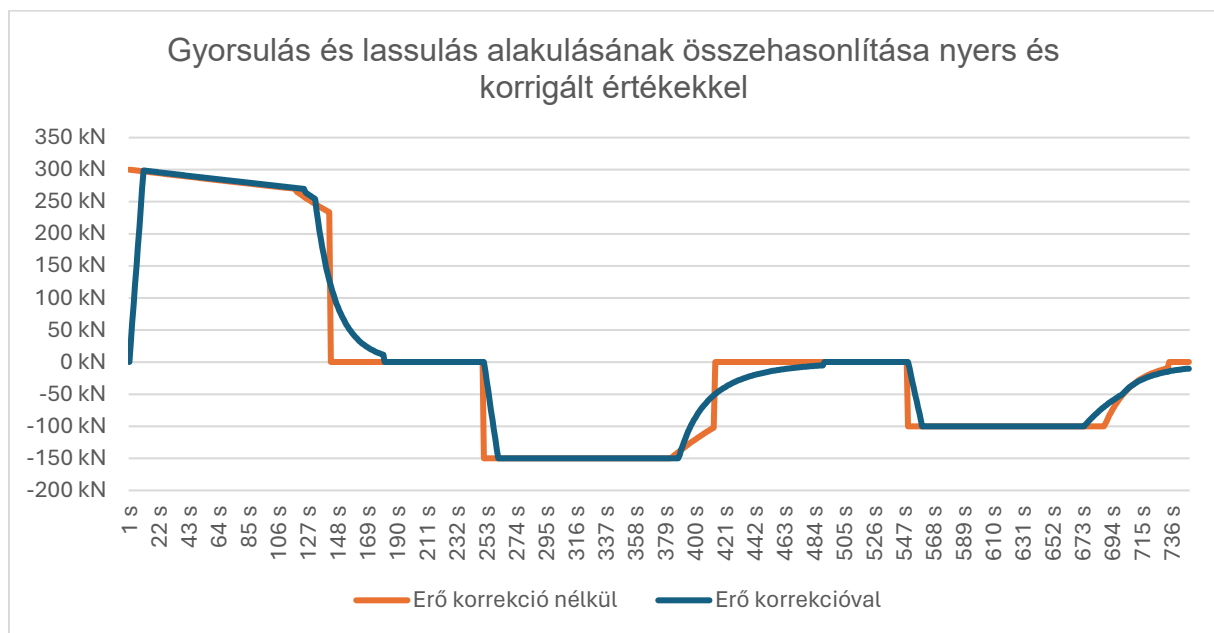
Mint ahogyan az 5.5. ábra mutatja, a szimuláció számára az alapértelmezett gyorsulás vagy lassulás mindig az elérhető maximális erővel történne meg, már a 0. időpillanattól is. Ez azt jelenti, hogy a meghatározott sebességre való felgyorsulás már a mozgás kezdetének pillanatában maximális gyorsulással valósulna meg. Természetesen ez a valóságban nem így történik, a vonóerő és ezáltal a gyorsulás egy bizonyos idő eltelte után képes csak a legnagyobb értéket elérni. A való életben számtalan külső körülmény befolyásolja a vonóerő megjelenésének intenzitását. A dolgozat témájába illeszkedve valós körülmények között az intenzitást szabályozni lehet a menetszabályzó kar ennek megfelelő kezelésével, amelyet a mozdonyvezető működtet (ennek modellezését az 5.8 fejezetben ismertetem). Természetesen ekkor is megjelenik egy szoftveres korlátozás, ugyanis a menetszabályzó 100%-ra történő beállításakor is eltelik egy bizonyos idő ameddig a jármű leadja a ténylegesen rendelkezésre álló vonóerőt.

A szimuláció során szükségét éreztem, hogy a valósághoz jobban közelítő mozgás valósuljon meg. Ennek érdekében bevezettem olyan korrekciós- és segédtevézőket, amelyekkel egy általános eljárást alakítottam ki a gyorsulások és lassulások erejének fokozatos megjelenésére. További eltérés az elméleti értékekhez képest, hogy a valóságban a mozdonyvezető a kívánt sebességet közelítve a vonóerő vagy fékerő nagyságát fokozatosan csökkenti, felkészülve a célsebesség komfortos elérésére és annak tartására. A dolgozatom során ezt az eseményt a *drive_control* eljárás hajtja végre, amelyre részletesen az 5.5 fejezetben térek ki. Az eljárás során egy AFB (automatikus menet-fék vezérlés) [8] üzemmódban közlekedő mozdony viselkedését szimuláltam, tehát a korrigálás nem a menetszabályzó vagy villamosfék controller képzeletbeli állásának módosításával, hanem a ténylegesen megjelenő erő módosításával történik. A valóságban ez történhet AFB üzemmódban, vagy manuális módon is.

Az elméleti (nyers) és korrigált gyorsulási értékek közötti eltérést az 5.6. ábra és az 5.7. ábra tartalmazza. A két ábrán jól látható, hogy a korrekció nélküli adatok nagyon pontos és gépies mozgást mutatnak. Az ábrán látható, végrehajtott forgatókönyv az 5.6 fejezetben kerül ismertetésre.



5.6. ábra: Az elméleti és a finomhangolt gyorsulási értékeken futó szimuláció sebességének változása (Forrás: saját ábra)



5.7. ábra: Az elméleti és a finomhangolt gyorsulási értékek görbéje a szimuláció futása során (Forrás saját ábra)

5.5 A drive_control eljárás

A *drive_control* programozási függvény a szimuláció egy szerves részét alkotja. A statikus forgatókönyv meghatározásánál ezt az eljárást hívom meg (lásd: 5.11. ábra), és ez szabályozza a jármű mozgását a futtatás során. A *drive_control* eljárás bemenő paraméterei a jármű egyedi azonosítója, célsebessége, össztömege és opcionálisan a menetszabályzó vagy a villamosfék szabályzó képzelt statikus állása.

Ezt az eljárást azzal a céllal hoztam létre, hogy lehetőségünk legyen bizonyos körülmények között bevezetni korrekciós tényezőket a gyorsulás és lassulás folyamatába.

Az eljárás meghívása során a függvény egy szelekciót követően eldönti, hogy a célsebesség eléréséhez gyorsuló vagy lassuló mozgást kell végezni. A szelekciót követően a program egy előltesztelő ciklusba lépve indítja el a mozgási folyamatot (5.10. ábra). Kiadja a jármű részére a *setSpeed()* parancsot a célsebességre, és elkezdődik a gyorsulás vagy lassulás. Ahogyan az előző alfejezetben említettem, korrekciós tényezőket határoztam meg a valóságosabb mozgás előállításához. Amint a *drive_control* eljárás kiadja a gyorsulási vagy lassulási parancsot, a jelenlegi beállítás szerint egy 10 másodperces időt határoztam meg, mialatt a gyorsulás vagy lassulás lineárisan éri el az elérhető legnagyobb értékét az aktuális sebesség alapján. Mielőtt a jármű elérné (mind gyorsulás, mind lassulás esetében) a kívánt célsebességet, a sebesség előtt 10 km/h-val megkezdődik egy korrekció, ahol egy szorzó tényezővel a célsebességhez közeledve arányosan csökkentésre kerül a vonóerő, és ez alapján a gyorsuló vagy lassuló mozgás. Ezt az 5.6. ábra és az 5.7. ábra szemlélteti, összehasonlítva a korrigálás nélküli futással.

A *drive_control* eljárásban lehetőségünk van modellezni a menetszabályzó és villamosfék szabályzó képzeletbeli állását. A valóságban a mozdonyvezető a menetszabályzóval tudja szabályozni a rendelkezésre álló vonóerőt 0-100% között. A villamosfékerő kontrollerrel a villamosfékerőt képes szabályozni összesen 9 (1A, 1B, 2, 3, 4, 5, 6, 7, VB) fokozatban. [8] A szimuláció során törekedtem ennek megvalósítására, így lehetőség van a menetszabályzó alapján történő vonóerő módosításra, valamint villamosfékerő szabályzó alapján villamosfékerő módosításra.

A fent említett tényezőket figyelembe véve a SUMO-nak átadott gyorsulást az 5.8. ábra, a lassulást az 5.7. ábra mutatja. A *szorzo_tenyezo* változóban az előzőekben említett, az adott helyzetben érvényes korrekciós tényező értéke van eltárolva.

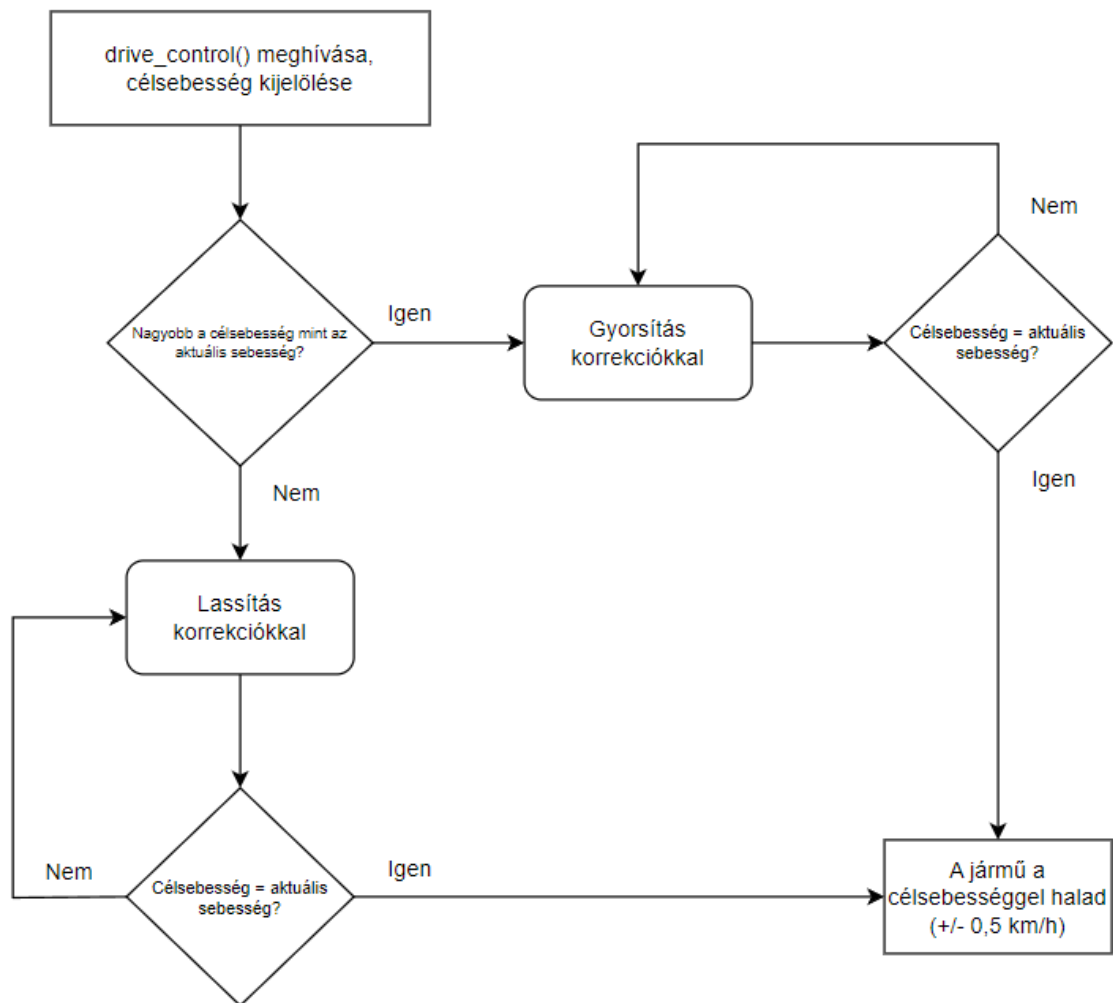
```
gyorsulas = (max_acceleration_of_1116(v, mass) * menetszabalyzo_allas) * szorzo_tenyezo
```

5.8. ábra: Gyorsulás meghatározása a *drive_control* programban (Forrás: saját kód)

```
lassulas = (max_deceleration_of_1116(v, mass) * dinamikus_fekszabalyzo_allas) * szorzo_tenyezo
```

5.9. ábra Lassulás meghatározása a *drive_control* programban (Forrás: saját kód)

A *drive_control* eljárás működését az 5.10. ábra mutatja be folyamatábra formájában.



5.10. ábra: A drive_control eljárás elméleti folyamatábrája (Forrás: saját szerkesztés)

5.6 Statikus forgatókönyvi elemek meghatározása

A szimuláció futtatásához meghatároztam egy alap forgatókönyvet. Ennek a forgatókönyvnek statikus elemei vannak, ezért, ha ugyanezen eseményeket reprodukáljuk eltérő tulajdonságú (jelen esetünkben eltérő össztömegű) szerelvényekkel, az eredmények ábrázolásával és elemzésével kirajzolódnak az egyszerűsített mozgási különbségek. A forgatókönyv a szimuláció elindításával indul a 0. másodperctől kezdve. Az alap forgatókönyv a következőképpen alakul:

- 0. másodperc és 50. másodperc között a szerelvény a pálya elején álló helyzetben van.
- Az 50. másodpercben kiadott *drive_control()* paranccsal 100 km/h (*target_speed* változó értéke) célsebességet állítunk be a szerelvényünknek.

- A 300. másodpercben kiadott *drive_control()* paranccsal 40 km/h célsebességre lassítjuk a szerelvényünket, csak villamosfék igénybevételével (pneumatikus féket nem számítom ki a szimuláció teljes egészében).
- A 600. másodpercben kiadott *drive_control()* paranccsal megállásra utasítjuk a szerelvényt. A későbbiekben láthatóvá válik, hogy miért nem teljesülhet a teljes tökéletes megállítás.

A kiadott *setSpeed()* parancs az előző fejezetben meghatározott aktuális maximálisan elérhető gyorsulás és lassulás értékével fog végrehajtásra kerülni. A statikus forgatókönyv programkódját az 5.11. ábra mutatja. Jelen dolgozatomban a szimulációk befolyásolását az ilyen, és ehhez hasonló műveletsorok valósítják meg.

```
if step < 50:
    traci.vehicle.setSpeed(veh_id, 0)
elif step == 50:
    drive_control(veh_id, target_speed, mass)
elif step == 300:
    drive_control(veh_id, 11.1111, mass)
elif step == 600:
    drive_control(veh_id, 0, mass)
```

5.11. ábra: Statikus forgatókönyv programkódja (Forrás: Visual Studio Code, saját kód)

Az 5.11. ábra a *step* változó a szimuláció aktuális idejét tartalmazza másodpercben, a *veh_id* változó a járművünk egyedi azonosítóját foglalja magában. A *drive_control* eljárásról bővebben az 5.4 fejezetben tettem említést, jelenlegi kontextusában ez utasítja a járművet a gyorsulás vagy lassulás végrehajtására. Az eljárás második argumentuma a célsebesség, amelyre a jármű fel fog gyorsulni. A *target_speed* változó 100 km/h sebességet jelent, a szimuláció számára megadva ez az érték 27,78 m/s. A *mass* változó a szerelvény össztömegét jelenti kg-ban megadva, jelenlegi esetben ez 1.423.000 kg, azaz 1.423 tonna.

5.7 Mozdonyvezetői attitűd megváltoztatása

A dolgozat jelenlegi állapotában a mozdonyvezetői attitűdöt a menetszabályzó és a villamosfék szabályzó állásának módosításával tehetjük meg. A szimuláció során a villamosfék kontroller kezelését is %-os értékben oldottam meg, ellentétben a valós fokozatoktól.

Amikor meghívjuk a *drive_control* eljárást, lehetőségünk van opcionálisan a két kontroller állását megadni. Alapértelmezetten mindkét kontroller állása 1, azaz 100%. Az értéket 0 és 1 között van lehetőségünk módosítani. Ezt úgy valósítottam meg a szimulációban, hogy a gyorsulás és lassulás kiszámításánál a kapott maximális értéket megszoroztam a képzelt szabályzó állásával százalékosan (lásd: 5.8. ábra és 5.9. ábra). Ennek értelmében a számított legnagyobb gyorsulás vagy lassulás egy bizonyos arányát adjuk csak át a SUMO részére, ezzel modellezve a szabályozó karok állását.

Ez az opció egy statikus szabályzó állást feltételez, tehát ha bemenetként adjuk meg az eljárásnak a kívánt értéket, akkor a gyorsulás vagy lassulás teljes egészében ezt a statikus szorzó tényezőt veszi figyelembe. Az 5.8 fejezetben bemutatom, hogy miképp lehet a szimuláció jelenlegi állapotában valamennyire dinamikusan változtatni a menetszabályzó vagy villamosfék kontroller állását.

5.8 Szabályzók állásainak dinamikus megváltoztatása

A kontrollerek állását lehetőségünk van egy gyorsulási vagy lassulási folyamaton belül is megváltoztatni. Ezt a lehetőséget jelenlegi állapotában a 6.2 fejezetben elvégzett valósággal való összehasonlítás miatt alakítottam ki. Jelenleg annak érdekében, hogy egy mozgási folyamaton belül változtassuk meg a képzelt szabályzók állásait, az 5.6. fejezetben szereplő statikus forgatókönyvhöz hasonló szelekciót kell készíteni. Az eljárás folyamán előre rögzített időpontokban történik a változtatás. Természetesen ez a valóságban nem ezen elvek mentén történik, azonban a valósággal való összehasonlítás érdekében ez egy megfelelő lehetőséget kínált az ellenőrzésre. A jövőben növelni szeretném ennek a lehetőségnek a rugalmasságát, hogy valóságosabb módon lehessen a dinamikus változtatást beépíteni a szimulációba.

6. A szimulációval elvégzett vizsgálatok

A 6. fejezetben azokat a vizsgálatokat mutatom be, amelyek végrehajtásához szükséges feltételeket az 5. fejezetben teremtettem elő. A szimuláció tényleges lefutási folyamatával a dolgozat során nem foglalkozom. Ennek oka, hogy a munkám során kizárólag a szimuláció kimenetéből származó adatokkal végeztem további számításokat és méréseket. Ezeket az eredményeket ebben a fejezetben mutatom be.

A szimuláció eredményei vesszővel tagolt fájlba (vagy más néven: .csv) kerülnek rögzítésre a szimuláció sikeres lefuttatása után. Ezeket a fájlokat például a Microsoft Excel segítségével táblázatos formában importálhatjuk, és diagrammokat készíthetünk belőlük. A diagramkészítésen túl számításokkal és adatelemzési eljárásokkal az eredményeket a számunkra megfelelő formátumhoz tudjuk igazítani. A nyersadatok másodpercenként érkeznek, és 4 oszlopban tartalmaznak értékeket.

A szimulációval többféle és többcélú vizsgálatot végeztem el a dolgozat készítése folyamán. Ezek közül néhány eredménye látható a korábbi fejezetekben (például az elméleti nyers és korrigált erők különbségének megjelenítése, 5.6. ábra). Ebben a fejezetben a több típus közül kettőre térek ki részletesen. Az egyik típus egy gyorsulási eltérést hivatott prezentálni (lásd: 6.1 fejezet), a másik típus pedig egy a valós mozdony felgyorsulásának összehasonlítása a szimulált mozdony felgyorsulásával (lásd: 6.2. fejezet).

6.1 Gyorsulással és lassulással végzett szimuláció

Ahogy a 3.3 fejezetben kitértem rá, a dolgozat elkészítése közben az ellenállásokat elhanyagoltam, nem vettem figyelembe. Ennek értelmében a képződött adataink olyan gyorsulási és lassulási körülményeket feltételeznek, ahol semmilyen környezeti ellenállás nem befolyásolja a jármű mozgását. Ezzel a típusú vizsgálattal azt szeretném meghatározni, hogy hogyan változik a vasúti járművek egyszerűsített mozgása különböző mozdonyvezetői attitűdök hatására. A mozdonyvezetői attitűdöt jelenleg a menetszabályzó és a villamosfékerő szabályzó képzelt állásának módosításával tudjuk befolyásolni. Ebben a fejezetben a menetszabályzó képzelt állásának két különböző, előre megadott értéket határoztam meg, és azt részletezem, hogy hogyan változtak a jármű gyorsulási paraméterei a két különböző esetben.

A gyorsulási és lassulási vizsgálat során kiszámítottam az egyszerűsített mozgások munkáját, amelyből meghatároztam a mozgás energiaszükségletét. A munka meghatározását a (12) képlettel, az energiaszükséglet meghatározását a (13) képlettel végeztem el.

$$W = \sum (F \times v) = F \times s \quad (12)$$

ahol:

- W : munka Joule-ban,
- F : vonóerő N-ban,
- v : sebesség m/s-ban.

$$E = \frac{W}{3,6 \times 10^6} \quad (13)$$

ahol:

- E : energiamennyiség kWh-ban,
- W : munka Joule-ban.

6.1.1 Esetek meghatározása

A sokoldalúbb vizsgálatok elvégzése érdekében meghatároztam két esetet, amely két eltérő tulajdonságú vasúti járművet jelent a gyakorlatban. Jelenleg a szimuláció nem foglalkozik a jármű általános paramétereivel (hossz, kocsik száma stb.), ezért az össztömeg módosításával deklarálhatunk eltérő típusú vonatokat.

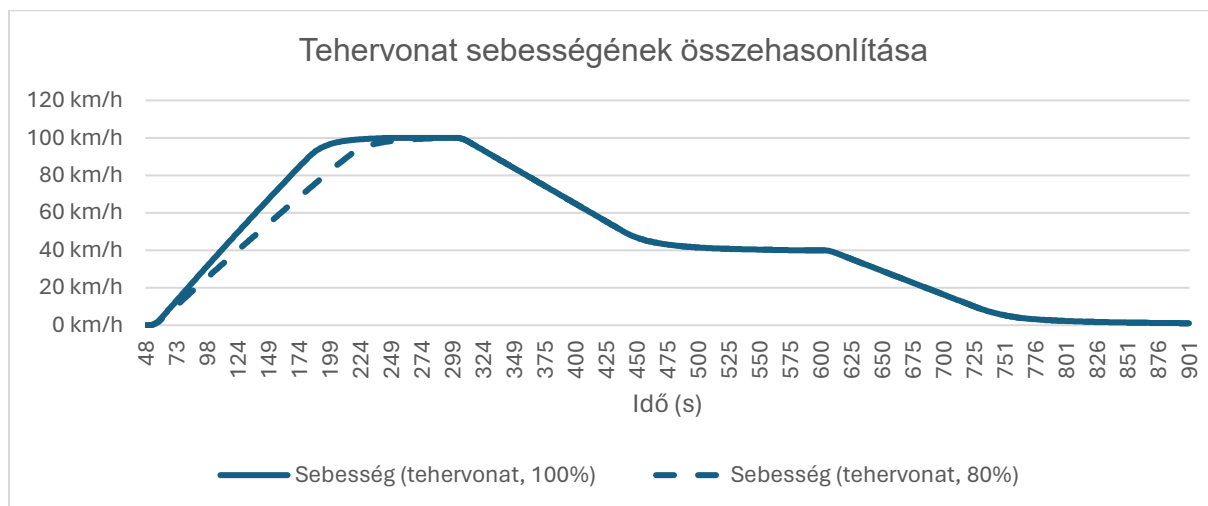
Az első esetben egy fiktív tehervonat fog az 5.6 fejezetben létrehozott statikus forgatókönyv alapján mozgást végezni. A tehervonat össztömege 1.423 tonna (1.423.000 kilogramm). A szimuláció futtatásra került először 100%-os menetszabályzó állással (a változat), majd 80%-os menetszabályzó állással (b változat).

A második esetben egy fiktív személyvonat fog mozgást végezni az 5.6 fejezetben meghatározottak alapján. A személyvonat össztömege 388 tonna (388.000 kg). A szimuláció futtatása hasonlóan az első esethez először 100%-os menetszabályzó állással (a változat), majd 80%-os menetszabályzó állással történt (b változat).

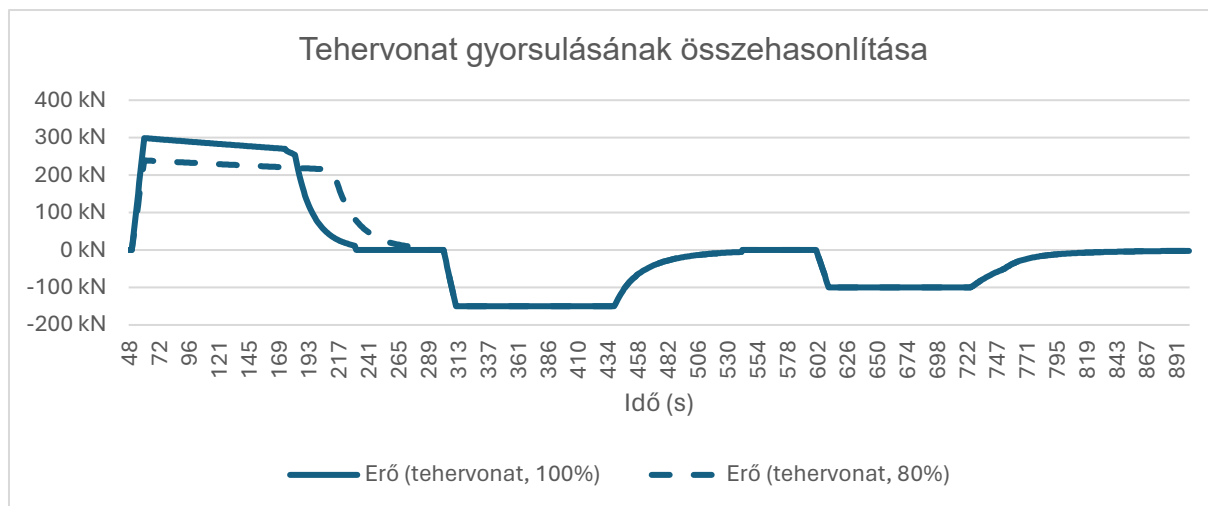
A menetszabályzó 100%-os állása a valóságban csak korlátozott esetekben alkalmazható, amely számos külső körülménytől függ. Ilyen körülmények lehetnek például a tapadási tényező a jármű kereke és a sín között vagy a felsővezeték hálózatba betápláló alállomás terheltsége miatti feszültségingadozások. Ezeket a külső okokat most figyelmen kívül hagytam, és feltételeztem, hogy a vonóerő 100%-os kihasználására a lehetőség adott. Mindkét eset futtatása során csak a menetszabályzó állását módosítottam, a villamosfék controller állásának konstans 100%-os (a valóságban *VB*, vagy *teljes villamosfék* [8]) értéket határoztam meg. Ennek eredményeképpen a két vizsgálat során eltérés csak a gyorsulási szakaszban lesz megfigyelhető. Korábban utaltam rá, hogy a megállás nem lesz tökéletes, csak közel 0 km/h sebességre fognak lassulni a vonatok. Ennek oka, hogy a villamosfék nem képes a mozdonyt állóhelyzetig fékezni. Tényleges megállításra a pneumatikus féket is alkalmazni kellene. A szimuláció elkészítése során a pneumatikus fékezést nem modelleztem le.

6.1.2 Tehervonat mozgása – első eset

Az első esetben a tehervonatunk felgyorsul 100 km/h-ra, majd egy rövidebb ideig tartó sebességtartást követően először 40 km/h-ra, majd közel 0 km/h-ra lassul. A vizsgálat sebességre vonatkozó eredményeit a 6.1 ábra, míg a vonóerőre vonatkozó eredményeket a 6.2. ábra részletezi.



6.1. ábra: Tehervonat mozgásának sebességre vonatkozó eredményei (Forrás: szimuláció eredményei)



6.2. ábra: Tehervonat mozgásának gyorsulásra vonatkozó eredményei (Forrás: szimuláció eredményei)

Jól látható, hogy a b változat gyorsulása egy elnyújtottabb sebességgörbét ad eredményül, ennek oka, hogy a kívánt sebességi érték elérése több időt (és nagyobb távolságot) vett igénybe. Megfigyelhető továbbá a 6.2. ábra alapján, hogy a b változat legnagyobb vonóerője körülbelül 240 kN szemben a maximális 300 kN-nal.

A tehervonat mozgásának szimulálásakor három alapvető mutatószámot jelöltem ki, amelyekkel statisztikailag is összehasonlíthatóvá válnak az eredményeink. A gyorsulás eredményeit a 6.1. táblázat, a lassítás eredményeit pedig a 6.2. táblázat tartalmazza.

Változat	Felvett kWh	Gyorsulási szakasz nagysága	Gyorsulási idő
a) Teher - 100%	150,10 kWh	2934,66 m	181 s
b) Teher - 80%	150,25 kWh	3674,28 m	225 s
Különbség	+ 0,15 kWh	+ 739,62 m	+ 44 s

6.1. táblázat: Tehervonat mozgásának számított gyorsulási értékei (Forrás: szimuláció eredményei)

A gyorsítási eredményekből látszik, hogy a 80%-os menetszabályzó állással való gyorsítás során körülbelül 740 méterrel nagyobb távolságot kellett megtenni, és 44 másodperccel több időnek kellett elteltie a célsebesség eléréséhez. Összességében ez 0,15 kWh –val növelte az energiafogyasztást.

Változat	Leadott kWh	Lassulási szakasz	Lassulási idő
Teher (100 km/h → 40 km/h)	-152,33 kWh	4085,04 m	242 s
Teher (40 km/h → ~0 km/h)		1031,56 m	310 s

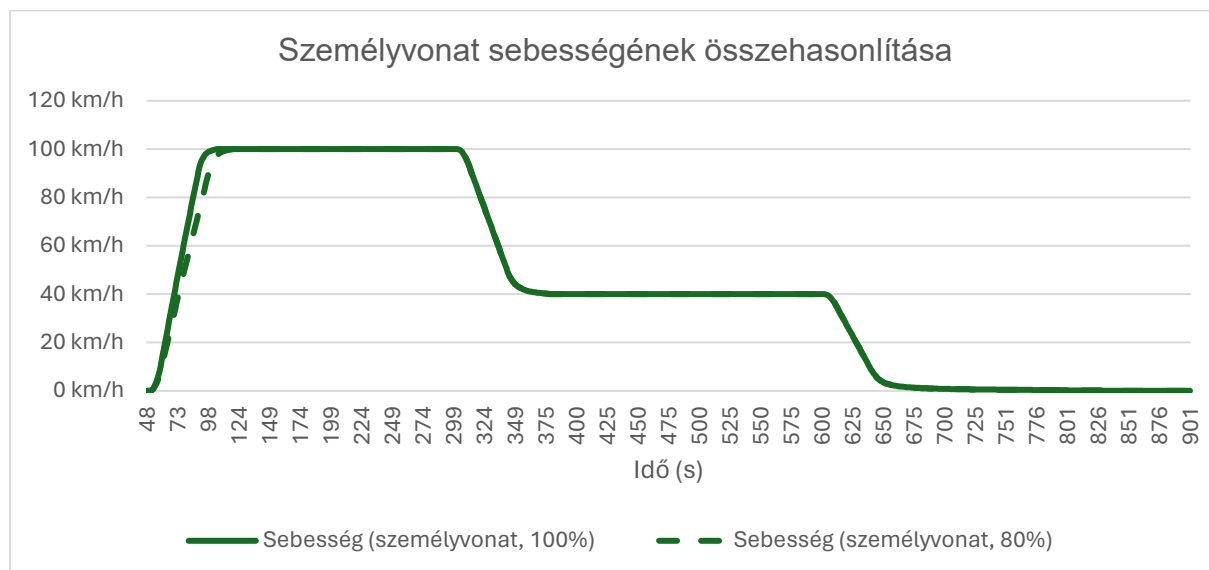
6.2. táblázat: Tehervonat mozgásának számított lassulási értékei (Forrás: szimuláció eredményei)

A lassulási eredmények alapján kijelenthető, hogy egy 1.423 tonna össztömegű vonattal nem lenne reális üzemi körülmények között 100 km/h-ról 40 km/h-ra lassítani.

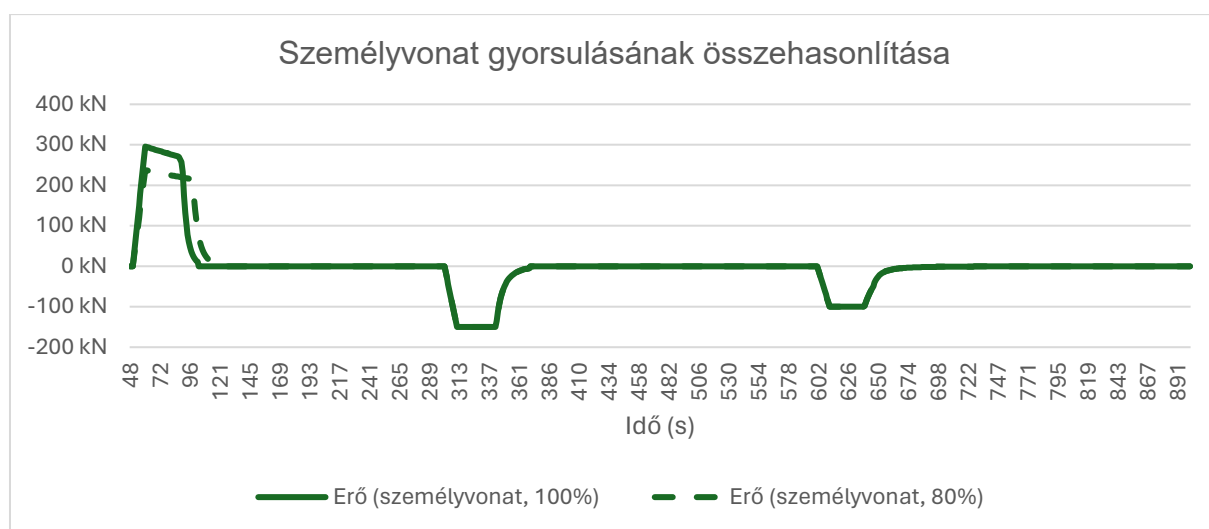
Ennek oka, hogy az alacsonyabb sebesség elérésére jellemzően 1 kilométer környéki távolságok állnak rendelkezésre. A 0 km/h körüli sebességre lassulás kizárólag villamosfékkel üzemi körülmények között nem teljesen értelmezhető, ugyanis a villamosfékerő a sebesség csökkenésével szintén csökken (lásd: 2.3. ábra), ezért egy bizonyos sebesség elérése után a lassulás meglehetősen alacsony. A 310 másodperces idő az első 1 km/h alatti sebesség elérésének az időpontja, a jármű jóval kevesebb idő alatt érte el az 5-10 km/h tartományt.

6.1.3 Személyvonat mozgása – második eset

A személyvonat forgatókönyve teljes egészében megegyezik a tehervonatével. A sebességi eltéréseket a 6.3. ábra, míg a vonóerő eltérését a 6.4. ábra tartalmazza.



6.3. ábra: Személyvonat sebességére vonatkozó eredményei (Forrás: szimuláció eredményei)



6.4. ábra: Személyvonat gyorsulására vonatkozó eredményei (Forrás: szimuláció eredményei)

A tehervonat grafikonjához képest szembetűnő, hogy meredekebb görbéket láthatunk. Ennek oka, hogy a vonat alacsonyabb össztömeggel közlekedett, ezért a gyorsulási- és lassítási értékei a tehervonatével megegyező nagyságú vonóerő kifejtése mellett jóval nagyobbak voltak. Már a sebességi ábrán is látható, hogy ebben az esetben a b változat kevesebb eltérést okozott a gyorsulási mozgás folytán. A 80%-os menetszabályzó állásból 201,20 méterrel több útra, és 12 másodperccel több időre volt szükség a célsebesség eléréséhez, mindez 0,17 kWh többlet energiafelvétellel járt.

Változat	Felvett kWh	Gyorsulási szakasz nagysága	Gyorsulási idő
a) Személy - 100%	40,30 kWh	812,40 m	54 s
b) Személy - 80%	40,47 kWh	1013,60 m	66 s
Különbség	+ 0,17 kWh	+ 201,20 m	+ 12 s

6.3. táblázat: Személyvonat mozgásának számított gyorsulási értékei (Forrás: szimuláció eredményei)

A személyvonat lassítási mozgásának mutatószámait a 6.4. táblázat tartalmazza. A 100 km/h-ról 40 km/h-ra lassulásra körülbelül 1247 méterre volt szükség kizárólag villamosfék használatával. Valós üzemi körülmények között bizonyos esetekben ez megvalósítható lehet, tekintve, hogy az általános fékúttávolság 1000 méter, az észlelési pont pedig 200 méterre található az általános fékúttávolság kezdeténél elhelyezett lassításra okot adó jelzéstől. Összegezve tehát, ha a mozdonyvezető már 300 méterrel korábban észleli a 40 km/h lassúmenetre előjelzést adó jelzőeszközt és azonnal megkezd a teljes villamosfék pozícióban való lassulást, akkor a szimuláció eredménye alapján a 40 km/h-val járható pályaszakasz elejéig sikerülne elérnie a célsebességet.

Változat	Leadott kWh	Lassulási szakasz	Lassulási idő
Személy (100 km/h → 40 km/h)	-41,86 kWh	1246,92 m	71 s
Személy (40 km/h → ~0 km/h)		327,69 m	87 s

6.4. táblázat: Személyvonat mozgásának számított lassulási értékei (Forrás: szimuláció eredményei)

A 40 km/h-ról közel 0 km/h-ig való lassulás esetén a körülbelül 328 méteres féktávolság lehetőséget adhat a mozdonyvezetőnek ilyen össztömegű vonattal kizárólag villamosfékkel az 5-10 km/h tartományba lassítani. A probléma hasonlóan a tehervonat esetéhez a megállításkor jelentkezik. A villamosfék önmagában nem képes a szerelvényt állóhelyzetre fékezni, 0-10 km/h környékén szükséges a pneumatikus féket is használni a teljes megállításhoz.

6.1.4 Az esetek vizsgálatának eredménye

A gyorsulással és lassulással végzett vizsgálatok eredményei összességében azt mutatták, hogy ha a gyorsulás intenzitása nagyobb, akkor a gyorsuláshoz szükséges energia alacsonyabb. Ennek értelmében a gazdaságos vezetés során – a külső körülmények adta lehetőségek figyelembevételével – törekedni kell a célsebesség mihamarabbi elérésére, ehhez pedig intenzív gyorsítást alkalmazni.

Természetesen a valós üzemi körülmények között ezt általánosságban nem lehet minden gyorsítási folyamatra érvényesíteni. Ennek alapja, hogy számos külső tényező befolyásolhatja (például fázishatárok, közeledő lassúmenetek) a jármű mindenkor legnagyobb megengedett sebességét. Azonban azt a szimuláció jelenlegi eredményei alátámasztották, hogy az intenzívebb gyorsulás energiahatékonyabb a kisebb volumenű gyorsítással szemben.

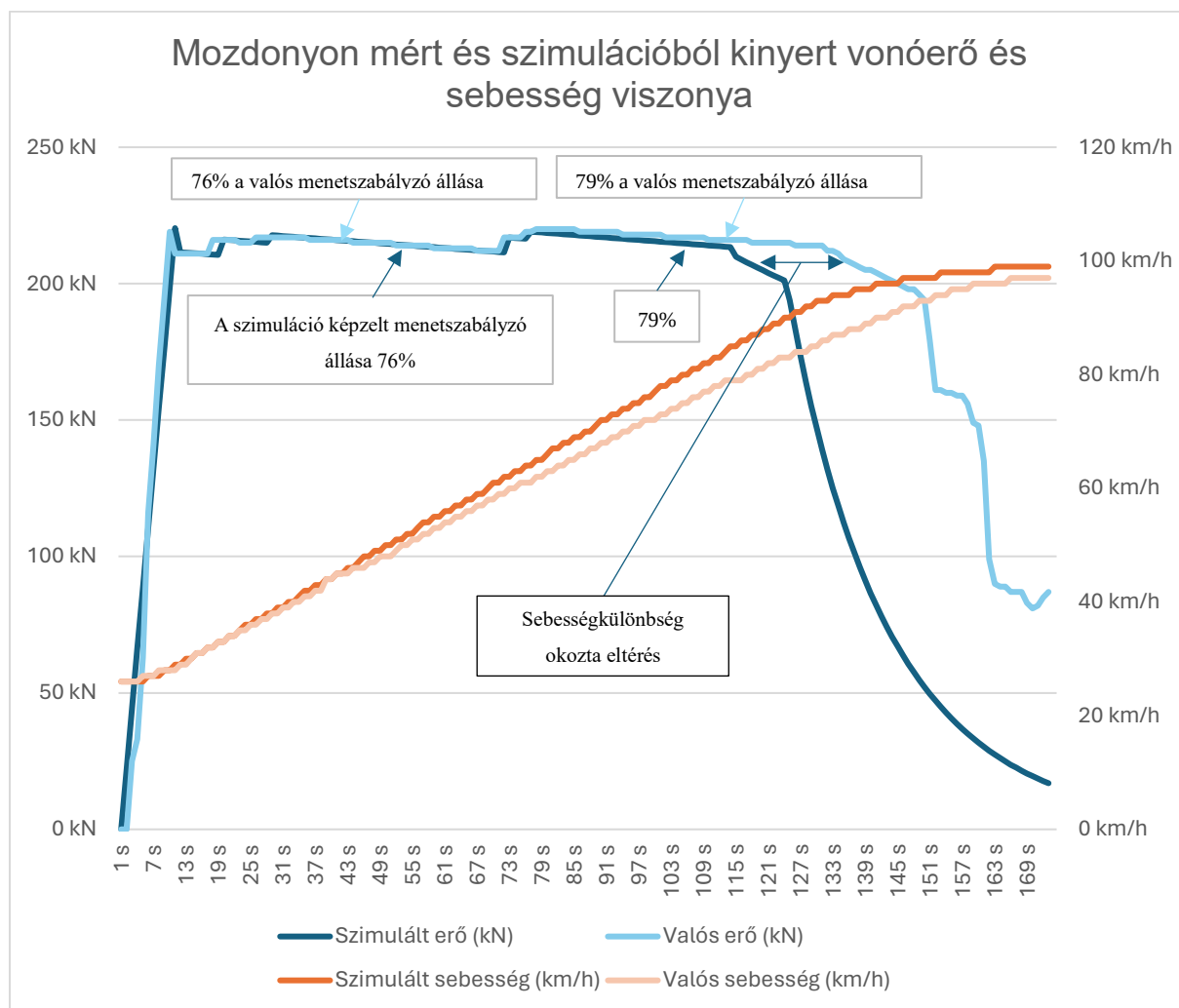
6.2 Összehasonlítás egy valós példával

Ebben a fejezetben részletesen ismertetek egy összehasonlítást, melynek célja a szimuláció eredményeinek ellenőrzése a valós adatokkal. Fontos rögzítenem, hogy a szimuláció a korábban említetteknek megfelelően nem tartalmazza az ellenállásokat, így egy teljes egészében elméleti közegben mozog. A szimulált vonat paramétereit beállítottam a valós vonattal megegyezőre, ami gyakorlati értelemben az össztömeg beállítását jelenti. Az össztömeg 1423 tonna, vagyis 1.423.000 kilogramm. A szimuláció lefuttatásához szükségem volt egy új forgatókönyv meghatározására, amellyel teljes egészében leképeztem a valóságban történt események időrendiségét. A valós vonatadatok, amelyeket rögzítettem a következők voltak:

- Pontos idő (óra:perc:másodperc),
- Sebesség (km/h-ban),
- Menetszabályzó állása (%),
- Vonóerő (kN).

Ahhoz, hogy a szimulációban reprodukálni tudjam a valós eseményeket, a menetszabályzó állását kellett a valóságnak megfelelő időrendben szabályoznom. A szimuláció a menetszabályzó állásának megfelelően biztosította a vonóerőt, valamint feltételezett egy 100 km/h-ra beállított AFB üzemmódot, tehát élt a valósághú korrekciókkal. Ezáltal a szimuláció futtatása során bemenetként szolgált a kontroller aktuális állása, kimenetként pedig az idő, sebesség és a vonóerő.

A két adatforrásból származó adatokat egy Excel munkalapon összefűztem, és egységes formátumba hoztam. Ezt követően készítettem el az összehasonlító diagrammot (lásd: 6.5. ábra).



6.5. ábra: Valós és szimulált adatok összehasonlítása (Forrás: saját mérés és szimuláció eredménye)

6.2.1 A valós példával történő összehasonlítás eredménye

A sebességi adatokból megállapítható, hogy a szimuláció sebessége a kezdetben közel azonos a valós sebességgel, azonban a sebesség növekedésével a valóságban csökken a gyorsulás intenzitása, ezáltal fokozatosan egyre több lesz a különbség a két adatsor között. Véleményem szerint ennek oka a menetellenállások megjelenése, amelyek a valóságban exponenciálisan növekednek a sebesség növekedésével. Ez megmagyarázza, hogy változatlan erő kifejtés mellett miért csökken a gyorsítás intenzitása.

A vonóerőknél jól kivehető, hogy a valós és a szimulált erők szinte teljes egészében megegyeznek egymással, addig a pontig, ameddig a sebességi értékek nem kezdenek el

nagyobb mértékben eltérni egymástól. Ez a pont körülbelül a 80. másodpercnél található, ahol a szimuláció sebessége továbbra is lineárisan növekszik, viszont a valóságban egyre növekedik a menetellenállás, amely exponenciálisan hat a gyorsulásra. A 114. másodpercnél figyelhető meg, hogy a szimuláció vonóerője elkezdi az exponenciális csökkenést a konstrukciónak megfelelően, viszont ugyanekkor a valós mozdony vonóerője még továbbra is valahol a lineáris tartományban fut. Az eltérés oka az, hogy a valós mozdony sebessége még nem érte el a 85 km/h-t, tehát nem kezdett még el csökkenni az elérhető vonóerő az exponenciális szakasznak megfelelően. Szintén a vonóerőknél látható, hogy a szimuláció esetén egy határozott gépi leszabályozás jelenik meg a célsebesség elérése előtt, addig a valós mozdonyon a korrekciót a mozdonyvezető manuálisan végezte a menetszabályzó kar állásának változtatásával.

A diagram alapján kijelenthető, hogy a szimuláció képes jól közelíteni egy valós vontatójármű mozgását, egyszerűsített formában. Kijelenthető továbbá, hogy a leképzett vonóerőgörbe meglehetősen pontosan képes visszaadni a mozdony valós vonóerőjét.

7. Lehetőségek a továbbfejlesztésre

A szimuláció jelenlegi, dolgozatban bemutatott állapotában képesek vagyunk előre meghatározott események mentén modellezést végezni, és reprodukálni a valós mozgásokat egyszerűsített formában (lásd: 6.5 ábra). Ennek eredményeképpen végezhetünk közelítőlegesen energiafogyasztási és menetidőbeli vizsgálatokat eltérő paraméterekkel futtatott szimulációkon keresztül, mint például egy új lassúmeneti szakasz megjelenése. Ezt az információt felhasználhatjuk például a menetrendi időadatok korrekciójára. Fontos, hogy a pneumatikus fékezés képességének hiánya miatt elsősorban olyan esetekben alkalmazható a szimuláció, ahol lehetőségünk adódik csak villamosfékkel lassítani az alacsonyabb sebességgel járható szakaszra.

A továbbfejlesztést tekintve elsődleges célom az ellenállások figyelembevétele a szimuláció során. Ennek elérése érdekében lehetőségem van mechanikai számításokkal részletesen meghatározni a vonat összes ellenállását, és ebből képezni egy függvényt. Másik megoldásként lehetőségem van a pályavasutak által általánosan meghatározott ellenállásfüggvényeket érvényesíteni. Az általánosítást tekintve előnyösebbnek tűnik az utóbbi megoldás, mivel további vontatójárművek leképzése esetén rugalmasabban lehet érvényesíteni az ellenállásokat.

További célom, hogy a pneumatikus fék hatását is figyelembe tudjuk venni a szimuláció során, ehhez pedig le kell modelleznem a levegős fékrendszer működését. A vasúton minden vonat közlekedése során fékszámításokat kell végezni, amely megfelelő eredménye alapfeltétele a járművek vonatban való közlekedésének. A fékszámítások alapján előállíthatónak tűnik a használatából eredendő maximális lassulás, amelyet már kiegészíthetünk a villamosfékezéssel.

A jelenlegi állapot egyik hátránya, hogy a szimuláció futtatásához forgatókönyvszerű közlekedésre van szükség. Előre megadott időpillanatokon történik például a menetszabályzó kar képzelt állásának megváltoztatása, vagy a fékezés megkezdése. A jövőben olyan formában szeretném kihasználni a SUMO alapkörnyezet és a TraCI által nyújtott lehetőségeket, hogy a lehető legkevesebb eseményt kelljen időben előre meghatározni. A célom az, hogy a lassítási és gyorsulási eseményeket a külső körülmények (következő jelző állása, lassúmeneti pályaszakasz, megállási pont) határozzák meg. Ha ez megvalósulna, akkor az alacsonyabb sebességgel járható szakaszra lassulás menetét nem kellene előre megadni, hanem a körülmények változásával a modell magától hajtaná végre a lassulást, a számára megadott vezetési elvek mentén.

Ábrajegyzék

2.1. ábra: Az ÖBB 1116-os Taurus mozdonya (Forrás: saját fénykép)	14
2.2. ábra: Egy fiktív jármű vonóerőgörbéje (Forrás: saját készítés)	16
2.3. ábra: Az ES64U2 mozdony vonó- és villamosfékerő görbéje [9].....	17
3.1. ábra: Vasúti ellenállások csoportosítása [11].....	24
3.2. ábra: A mozdony vonó- és villamosfékerő görbéje a leképzés során kiszámított értékekkel ábrázolva (Forrás: saját ábra)	25
5.1. ábra: Elérhető gyorsulás megadása a jármű számára (Forrás: saját szerkesztés).....	29
5.2. ábra: Vonóerő és gyorsulás visszaadásának programkódja (Forrás: saját kód).....	30
5.3. ábra: Elérhető lassulás kiszámítása a jármű számára (Forrás: saját szerkesztés).....	31
5.4. ábra: Villamosfékerő visszaadása programkódban (Forrás: saját kód)	32
5.5. ábra: Mozdony gyorsulása és lassulása maximális vonóerővel és villamos fékerővel korrekció nélkül (Forrás: saját szerkesztés)	32
5.6. ábra: Az elméleti és a finomhangolt gyorsulási értékeken futó szimuláció sebességének változása (Forrás: saját ábra).....	34
5.7. ábra: Az elméleti és a finomhangolt gyorsulási értékek görbéje a szimuláció futása során (Forrás saját ábra).....	34
5.8. ábra: Gyorsulás meghatározása a drive_control programban (Forrás: saját kód).....	35
5.9. ábra Lassulás meghatározása a drive_control programban (Forrás: saját kód).....	35
5.10. ábra: A drive_control eljárás elméleti folyamatábrája (Forrás: saját szerkesztés)	36
5.11. ábra: Statikus forgatókönyv programkódja (Forrás: Visual Studio Code, saját kód).....	37
6.1. ábra: Tehervonat mozgásának sebességre vonatkozó eredményei (Forrás: szimuláció eredményei).....	41
6.2. ábra: Tehervonat mozgásának gyorsulásra vonatkozó eredményei (Forrás: szimuláció eredményei).....	42
6.3. ábra: Személyvonat sebességére vonatkozó eredményei (Forrás: szimuláció eredményei)	43
6.4. ábra: Személyvonat gyorsulására vonatkozó eredményei (Forrás: szimuláció eredményei)	43
6.5. ábra: Valós és szimulált adatok összehasonlítása (Forrás: saját mérés és szimuláció eredménye).....	46

Irodalomjegyzék

- [1] Aradi Sz., Bécsi T. (2009) *Szubmikroszkopikus forgalomszimulációs keretrendszer tervezési kérdései*, Innováció és fenntartható felszíni közlekedés IFFK-KONFERENCIA, 2009
- [2] PTV Vissim weboldala, <https://www.ptvgroup.com/en/products/ptv-vissim>, 2024.03.22.
- [3] PTV Visum weboldala, <https://www.ptvgroup.com/en/products/ptv-visum>, 2024.03.22.
- [4] OpenTrack weboldala,
http://www.opentrack.ch/opentrack/opentrack_e/opentrack_e.html#Events, 2024.03.26.
- [5] Lopez, A. P. - Behrisch, M. – Bieker-Walz, L. – Erdmann, J. – Flötteröd, Y. – Hilbrich, R. – Lücken, L. - Rummel, J. – Wagner, P. – Wießner, E. (2018) *Microscopic Traffic Simulation using SUMO*, IEEE Intelligent Transportation Systems Conference (ITSC), 2018.
- [6] Gudwin, R., R. (2016) *Urban Traffic Simulation with SUMO*, Copyright © 2016 DCA-FEEC-UNICAMP, p. 11
- [7] TraCI – SUMO Dokumentációja, <https://sumo.dlr.de/docs/TraCI.html>, 2024.03.26.
- [8] *Kezelési utasítás az ES64U2 villamosmozdony sorozathoz (2002).*, Doku-Nr.: F0489.38.1_020701 p.
- [9] Benedek T., Szabó A., Iványi Z. (2012) *Vasúti jármű mérés technika*, Egyetemi tananyag, p. 110, Budapest
- [10] Arató K. (2008) *Vasúti rendszertechnikai ismeretek*, Egyetemi tankönyv, pp. 10-11
- [11] Dr. Kazinczy L. (2004) *Vasúti pályák*, Egyetemi segédlet, p. 42
- [12] SUMO-Workshop (2019), Berlin – Deutsche Bahn – DigitalBase, 2019. szeptember 5.
- [13] Interfacing TraCI from Python,
https://sumo.dlr.de/docs/TraCI/Interfacing_TraCI_from_Python.html, 2024.03.29.