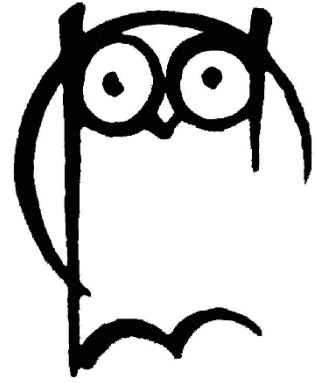


Debreceni Egyetem

Műszaki Kar

Műszaki menedzser Tanszék



Ipari folyamatok szimulációja digital lean eszköz alkalmazásával

Szerző: Tóth Szabolcs Balázs MSc hallgató

Témavezető: Dr. Budai István

Tanszékvezető: Dr. T. Kiss Judit

Debrecen

2023.

Összefoglaló

Ez a pályamunka egy ipari 3D-s környezet fejlesztését mutatja be a Godot játékmotorban, melynek folyamatvezérléséért a Codesys-t választottam TCP/IP kapcsolaton keresztül. A projekt célja, hogy bemutassa ennek a megközelítésnek a megvalósíthatóságát és előnyeit a mérnöki menedzsmentben, különösen az ipari folyamatok szimulációjára és irányítására vonatkozóan. A folyamatok szimulációjának fontosságát nem lehet eléggé hangsúlyozni, mivel lehetővé teszi a mérnökök számára a rendszerek biztonságos és költséghatékony tesztelését és optimalizálását. A folyamatszimuláció lehetővé teszi az esetleges problémák azonosítását és megoldását, mielőtt azokat valós környezetben implementálnák. Ez a megközelítés csökkentheti a költségeket, minimalizálhatja a kockázatokat, és rendszeres fejlesztéshez vezethet.

A piacon különböző szimulátorok állnak rendelkezésre, mindegyiknek megvannak a maga erősségei és korlátai. A népszerű szoftverek közé tartozik például a Factory I/O. A megfelelő szimulátor kiválasztása olyan tényezőktől függ, mint a kívánt részletesség, a könnyű kezelhetőség, az integrációs képességek és az iparág sajátos igényei.

A Fischertechnik, egy építőjáték-rendszer, amelyet az oktatásban és az iparban is használnak komplex mechatronikai rendszerek szimulálására és prototípusok készítésére. Sokoldalúsága, modularitása és különböző vezérlőrendszerekkel való kompatibilitása vonzóvá teszi az oktatási intézmények, kutatólaboratóriumok és ipari alkalmazások számára.

Összefoglalva a projekt célja a Godot játékmotor és a Codesys vezérlőszoftver integrálása egy funkcionális és hatékony szimulációs környezet létrehozása érdekében. A Godot nagy teljesítményű, nyílt forráskódú platformot kínál a 3D modellezéshez és vizualizációhoz, míg a Codesys átfogó eszközkészletet biztosít az automatizálási rendszerek programozásához és vezérléséhez. A szakirodalmi áttekintés tárgyalja az e technológiák integrálásával kapcsolatos lehetséges kihívásokat és megoldásokat.

Summary

This project presents the development of an industrial 3D environment in the Godot game engine, for which I chose Codesys for process control over TCP/IP. The aim of the project is to demonstrate the feasibility and benefits of this approach in engineering management, in particular for the simulation and control of industrial processes. The importance of process simulation cannot be overemphasized as it allows engineers to test and optimize processes in a safe and cost-effective manner. Process simulation allows potential problems to be identified and solved before they are implemented in a real environment. This approach can reduce costs, minimise risks and lead to continuous process improvement.

Different process simulators are available on the market, each with its own strengths and limitations. Popular simulators include Factory I/O. The choice of the right simulator depends on factors such as the level of detail required, ease of use, integration capabilities and the specific needs of the industry.

Fischertechnik, a building-block system, is used in education and industry to simulate and prototype complex mechatronic systems. Its versatility, modularity and compatibility with different control systems make it attractive for educational institutions, research laboratories and industrial applications.

In summary, the project aims to integrate the Godot game engine, Codesys control software and Fischertechnik components to create a functional and efficient simulation environment. Godot offers a powerful open source platform for 3D modelling and visualisation, while Codesys provides a comprehensive set of tools for programming and controlling automation systems. The literature review discusses potential challenges and solutions for integrating these.

Tartalomjegyzék

Összefoglaló	2
Summary.....	3
Tartalomjegyzék	4
Bevezetés	6
1 Ipari folyamatok.....	7
1.1 Ipari gyártási folyamatok fajtái.....	7
2 Lean gyártás	8
2.1 Az öt elsődleges alappillér leírása	8
2.2 Hatékony veszteségsökkentő gyártás	10
3 Digital Lean	11
3.1 A digital lean gyártás megjelenése	11
3.2 Digitális hulladék.....	12
3.3 A kiber-fizikai termelési rendszerekre (CPPS) való áttérés: A termelés jövője... 12	
4 Ipari folyamatok szimulációja.....	14
4.1 Szimuláció fajtái	15
4.1.1 Diszkrét esemény szimuláció	15
4.1.2 Folyamatos szimuláció	16
5 Folyamatszimulációk alkalmazása az iparban	17
5.1 Esettanulmány.....	17
5.1.1 Autóipar (karosszéria gyártás).....	17
6 Anyag és módszertan	21
6.1 Felhasznált eszközök	22
6.1.1 Fischertechnik színválogató folyamat modell	22
6.1.2 Factory I/O.....	25

6.1.3	Godot	26
6.1.4	Shapr3D	26
6.1.5	CODESYS	28
6.1.6	PLC TCP/IP szerver	29
6.1.7	Godot TCP/IP kliens.....	31
6.2	Fischertechnik színválogató állomás bemutatása és elemzése	34
6.2.1	Technikai adatok.....	34
6.2.2	Gyári program és működés.....	36
6.2.3	Válogatási idő	38
6.3	A válogató állomás megépítése és vezérlése digital lean szoftverben.....	40
6.3.1	Factory I/O-s megvalósítás	41
6.3.2	Saját szimulációs megvalósítás	43
6.3.3	Vezérlő logika újragondolása	43
	Összegzés	47
	Irodalomjegyzék	48

Bevezetés

A mai globális versenyhelyzetben egyre fontosabbá vált a költséghatékonyság és az ipari folyamatok optimalizálásának igénye. A Lean gyártás széles körben elfogadott megközelítésként jelent meg e célok elérésére, a hulladékcsökkentésre, a folyamatos fejlesztésre és az értékteremtésre összpontosítva. Bár a lean gyártás számos iparágban sikeresnek bizonyult, még mindig van hova fejlődni, különösen a folyamatszimuláció területén.

A szimulációs eszközök nélkülözhetetlenné váltak az ipari folyamatok optimalizálásában, lehetővé téve a mérnökök számára a folyamatok biztonságos és költséghatékony tesztelését és optimalizálását. Egy folyamat szimulálásával, mielőtt azt valós környezetben megvalósítanák, a potenciális problémák azonosíthatók és megoldhatók, így minimalizálhatók a kockázatok és csökkenthetők a költségek. Azonban nem minden szimulációs eszköz egyforma, és az adott alkalmazáshoz megfelelő eszköz kiválasztása olyan tényezők gondos mérlegelését igényli, mint a szükséges részletesség, a könnyű használhatóság, az integrációs képességek és az iparág sajátos igényei.

Ebben a dolgozatban egy ipari 3D-s környezet fejlesztését vizsgálom a Godot játékmotor és a Codesys segítségével, amely ígéretes megközelítést kínál a folyamatszimulációhoz. Mielőtt belemerülnénk a projekt sajátosságaiba, fontos egy lépést hátra lépni, és megvizsgálni az ipari folyamatok és a lean gyártás tágabb kontextusát. Átfogó irodalmi áttekintésen keresztül vizsgálom a lean gyártás elveit és gyakorlatát, előnyeit és a különböző iparágakban történő megvalósítását. Emellett feltárom a különböző szimulációs eszközöket és azok alkalmazását az ipari folyamatok optimalizálására.

A lean gyártás elveinek, a szimulációs eszközöknek és az innovatív technológiáknak a kombinálásával ez a dolgozat a folyamatszimuláció és a mérnöki irányítás újszerű megközelítését mutatja be.

1 Ipari folyamatok

Az ipari folyamatok a nehézipar jelentős részét képezik. Ezek kémiai, fizikai, elektromos és mechanikai lépéseket foglalnak magukban, amelyek a gyártási folyamatot segítik. Számos ipari folyamat létezik, amelynek célja egy termék előállítása vagy alakítása. Ilyen például az összetett, többbretű tevékenység, amely sokféle gépet, szerszámot és berendezést foglal magába. Az automatizálásnak is több szintje van, számítógépeket, robotokat, egyéb végrehajtókat vagy felhőalapú technológiákat használnak fel.

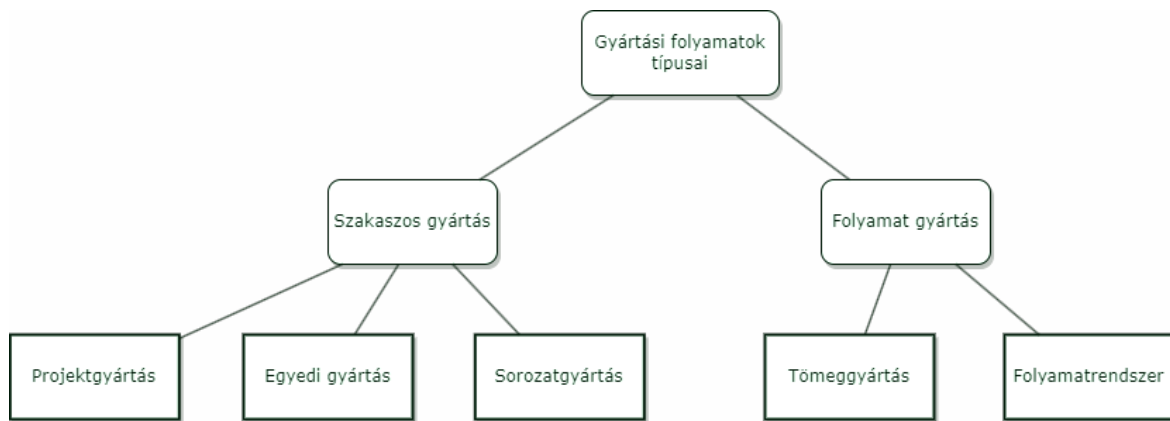
1.1 Ipari gyártási folyamatok fajtái

Szakaszos gyártás

A szakaszos gyártás esetében a gyártási rendszer alapvetően rugalmas, a gyártandó termékek köre folyamatosan változik, és a gyártási folyamatot teljes mértékben a vevői igények vezérlik. Ezeket a gyártási módszereket jellemzően a húzó gyártási filozófia szerint gyártandó termékek előállítására használják. Az ilyen típusú termelésre jellemző a speciális célú gépek és berendezések használata, a végtermékek széles választékának előállítása és a gyártandó termékek technológiájának változatossága, valamint az alacsony gyártási darabszám. Csak a vevők által megrendelt termékeket gyártják, így kis készletek halmozódnak fel [1].

Folyamatos gyártás

A folyamatos (más néven ismétlődő) gyártás viszonylag kis termékváltozatok nagyobb mennyiségben történő előállítása. Ezt a gyártási módszert jellemzően a toló termelési filozófia szerint gyártandó termékek előállítása során alkalmazzák. A technológiai folyamat minden egyes megmunkáló egysége a gyártandó termék technológiai műveleteinek sorrendjében van elhelyezve. Az egyes gyártási műveletek közötti anyagáramlást többnyire automatizált anyagáramlást biztosító berendezésekkel valósítják meg. A folyamatos gyártás számos előnnyel jár a szakaszos gyártással szemben: megkönnyíti a gyártástervezést és az ütemezést, standardizálja a termékeket és a gyártási folyamatokat, valamint jelentősen csökkenti a gyártási ciklusidőt, ezáltal javítja a termelés hatékonyságát. Jobban kihasználhatók a berendezések és az emberi erőforrások, és a gyártási folyamat kevesebb szakképzett munkaerővel is elvégezhető. A termékek fajlagos költsége a nagy volumenű gyártásnak köszönhetően alacsonyabb [1].



1. ábra: Gyártási folyamatok típusai [1]

2 Lean gyártás

Magát a "lean gyártás" kifejezést 1986-ban Krafcik alkotta meg, de nemzetközileg csak valamivel később vált ismertté a *The Machine That Changed the World* (A gép, amely megváltoztatta a világot) című könyv révén. A lean manufacturing a Toyota termelési rendszeréből (Toyota Production System, TPS) kölcsönzött gyártási szemlélet és filozófia, amelynek fő célja a vállalat veszteségeinek minimalizálása az emberek, a készletek, a szállítási idők, a gyártási folyamatok és a gyártási terület tekintetében. Ennek eredményeként jelentősen megnő a termelés rugalmassága és javul az ügyfélszolgálat, miközben segít a kifogástalan minőségű termékek minél hatékonyabb előállításában [2].

2.1 Az öt elsődleges alappillér leírása

A lean gyártás öt elsődleges eleme a következő: gyártási folyamat, szervezet, folyamatirányítás, mérőszámok és logisztika (1. táblázat). Ezek az elemek a stabil lean gyártási program támogatásához szükséges különböző aspektusokat képviselik, és ezeknek az elemeknek a teljes körű alkalmazása az, ami egy vállalatot a világszínvonalú gyártás felé vezető útra terel [3, 4].

Az alábbiakban az öt elsődleges elem mindegyikének alapvető meghatározása következik:

- Gyártási folyamat: A gyártási cella részeként végrehajtott fizikai változásokra és tervezési szabványokra utal [3, 4].
- Szervezet: Az alkalmazottak szerepének és funkcióinak meghatározására, az új munkafolyamatokra vonatkozó képzésre és a szervezeten belüli kommunikáció javítására összpontosít [3, 4].

- Folyamatirányítás: A gyártási folyamat nyomon követését, ellenőrzését, stabilizálását és folyamatos javítását foglalja magában [3, 4].
- Mérőszámok: A látható teljesítménymutatók mérésére és javítására összpontosít, beleértve a kitűzött eredményeket elérő csapatok/egyének jutalmazását és elismerését [3, 4].
- Logisztika: Magában foglalja a működési szabályok meghatározását, az anyagáramlás tervezését és ellenőrzését szolgáló mechanizmusok létrehozását az ellátási láncban [3, 4].

1. táblázat: Lean gyártás 5 alappilére [3]

Gyártási folyamat	Szervezet	Folyamatirányítás
• Termék/mennyiség felmérése (termékcsoport) • Folyamat feltérképezése • Útvonalelemzés (folyamat, munka, tartalom, mennyiség) • Takt számítások • Munkaterhelés kiegyensúlyozása • Kanban méretezés • Cellák elrendezése • Szabványos munka	• Termékközpontú, több szakterületet átfogó csapat • Lean menedzser fejlesztés • Képzés (lean-tudatosság, cellavezérlés, mérőszámok, folyamatos fejlesztés) • Kommunikációs terv • Szerepek és felelősség	• TPM • Poka-yoke • SMED • Grafikus munkautasítások • Vizuális ellenőrzés • Folyamatos fejlesztés • 5S
Mérőszámok		Logisztika
• Időben történő szállítás • Folyamat átfutási idő • Teljes költség • Minőségi hozam • Készlet • Helykihasználás • Útvonal távolság • Termelékenység		• Előrehaladási terv • Mix-modell gyártás • Megmunkálható munka • Kanban pull jelzés • A, B, C alkatrészek kezelése • Szervizcella-megállapodások • Ügyfél/szállító összehangolása • Működési szabályok

A lean gyártás öt elsődleges eleme teljes mértékben lefedik a lean gyártás megvalósítása során felmerülő problémák körét. Mindegyik elem egy adott területre összpontosít, és a

különböző tevékenységeket részekre bontja. Bár minden egyes elem önmagában is fontos a sikeres lean gyártási program bevezetéséhez, a lényeg mégis mind az 5 alkotóelem integrálásából fakad. Például a gyártási folyamat megteremti a változás alapjait. Az emberek látják a műhelyben folyó tevékenységet, a bútorok mozgatását, a gépek, a padló vagy a falak festését, a területek takarítását. Izgalom és energia övezi ezt a látható változást. Ehhez jönnek még a kevésbé látható változások az infrastruktúrában a szervezeti szerepek és felelőségek, az új munkamódszerek, a személyzet képzése, a többfunkciós csapatmunka, valamint a vevői/beszállítói kapcsolatok meghatározása tekintetében. Végezetül adjuk hozzá az üzemi mérések látható jeleit, a javítás alatt álló berendezéseket, a munkaállomásokon kifüggesztett grafikus munkautasításokat, valamint a gépátállási idők rögzítését és javítását. Ezek az elsődleges elemek kiegészítik egymást, és a sikeres megvalósítás részeként mindegyiknek támogatnia kell egymást [3].

2.2 Hatékony veszteségcsökkentő gyártás

A lean gyártás egy olyan folyamat, amely minden résztvevő részéről elkötelezettséget és odaadást igényel. Ez magában foglalja a jelenlegi folyamataink kritikus szemléletű vizsgálatát, olyan területek azonosítását, ahol megszüntethetjük a pazarlást, és olyan változtatások végrehajtását, amelyek segítenek céljaink elérésében. Ez az út nem könnyű, de szükséges, ha versenyképesek akarunk maradni a mai globális piacon.

A lean gyártás egyik legfontosabb alapelve a folyamatos fejlesztés koncepciója. Ez azt jelenti, hogy mindig keressük a módját, hogyan javíthatjuk folyamatainkat, csökkenthetjük a hulladékot és növelhetjük a hatékonyságot. A folyamatos fejlesztés kultúrájának elfogadásával egy rugalmasabb és alkalmazkodóképesebb gyártási környezetet hozhatunk létre, amely gyorsan tud reagálni a változó vevői igényekre.

Egy másik fontos szempont a dolgozók elkötelezettségének fontossága. Azáltal, hogy a dolgozókat bevonjuk a változtatási folyamatba, biztosíthatjuk, hogy mindenki részt vesz a program sikerében. Ez fokozott innovációhoz, jobb problémamegoldáshoz és a folyamat iránti nagyobb felelősségvállaláshoz vezethet.

Igazából ez a folyamat nem csak a hulladék csökkentéséről és a hatékonyság növeléséről szól. Hanem arról is, hogy értéket teremtsünk ügyfeleink számára. Az ügyfelek igényeire és elvárásaira összpontosítva olyan folyamatokat tervezhetünk, amelyek az ő igényeiknek megfelelő, kiváló minőségű termékeket és szolgáltatásokat nyújtanak. Ez a vevők elégedettségének, hűségének és visszatérő ügyfeleinek növekedéséhez vezethet.

3 Digital Lean

A lean fogalma alapvető szerepet játszik az ipari folyamatok hatékonyságának és eredményességének alakításában. A Toyota termelési rendszeréből kiindulva a lean elvek tradicionálisan az érték maximalizálására összpontosítanak a veszteségek minimalizálásával. A digitális technológiák fejlődése során ezek az elvek szerves részévé váltak az alapelvek továbbfejlesztésének. Ez az integráció egy új paradigmát hozott létre, amelyet "Digital Lean" néven ismerünk [5].

A Digital Lean nem pusztán egy divatos szó, olyan átalakító megközelítést képvisel, ahol a digitális eszközöket a műveletek racionalizálására, a termékminőség javítására és a vevői elégedettség fokozására használják fel. Az olyan technológiák, mint a dolgok internete (IoT), a mesterséges intelligencia (AI), a gépi tanulás és az adatelemzés, beépítésével a Digital Lean új szintre emeli a lean gyártás alapfilozófiáját [5].

Ahogy az ipar különböző területei dinamikusabbá és a vevői igények összetettebbé váltak, az agilis és adat vezérelt gyártási környezet iránti igény kiemelkedően fontossá vált. A Digital Lean ezt az igényt a digitális megoldásoknak a lean keretrendszerbe történő integrálásával elégíti ki, lehetővé téve a szervezetek számára, hogy gyorsan reagáljanak a változó piaci körülményekre és a belső folyamatok dinamikájára [5].

3.1 A digital lean gyártás megjelenése

Az ipari termelés folyamatosan fejlődő világában a digitális technológiák megjelenése a lean gyártás átalakuló korszakát hozta el. Ez az új szakasz, amelyet Digital Lean Manufacturing (DLM) néven ismerünk, a hagyományos lean elveket a fizikai téren túlmutatva a digitális innováció erejével ruházza fel. A lean gyártás lényege mindig is a pazarlás kiküszöbölése és a folyamatok állandó javítása volt. A digitális technológiák integrálásával ez a koncepció felértékelődött, lehetővé téve a "physical-to-digital" átalakítást, amely mélyrehatóbb és hatásosabb változást eredményez. Ennek a megközelítésnek a középpontjában a kiber-fizikai termelési rendszerek (CPPS) állnak, amelyek az intelligens gyárak működésének a minimális fizikális és digitális hulladék elérése felé történő orientálásában játszanak központi szerepet. Ez a cél egy szinkronizált termelési környezet révén érhető el, beleértve a digitális ikrek használatát is. Ezek a virtuális modellek és szimulációk, amelyek a virtuális gyárat reprezentálják, elengedhetetlenek a hulladékmentes termelési műveletek tervezéséhez, ellenőrzéséhez és validálásához a kibervilágban, mielőtt azok a valós gyártásban megvalósulnának. Ezenkívül ezek a rendszerek javíthatják a valós idejű

teljesítmény felügyeletét a fizikai világban. Ez a nyomon követés létfontosságú annak értékeléséhez, hogy a termelési műveleteket a lehető legmagasabb szintű produktivitással és minőséggel hajtják-e végre a terveknek megfelelően, vagy még mindig vannak lehetőségek a javításra és a folyamatos fejlesztésre (Kaizen). A digitális és smart gyártási technológiák elfogadása megkönnyíti a "digitize-to-connect" megközelítést. Ez a megközelítés jelentős a vertikális és horizontális gyári nagy-adatintegráció megvalósítása szempontjából, hozzájárulva a hulladék minimalizálásához a digitális lean képességek kialakításának folyamatában. Ezek a képességek magukban foglalják az adatok átláthatóságát, amelyet az új generációs Andon rendszerek és a digitális műszerfalak példáznak [5].

3.2 Digitális hulladék

A lean hagyományosan úgy jellemzi a hulladékot, mint minden olyan tevékenység, amely nem teremt hozzáadott értéket. A Merriam-Webster szótár meghatározása szerint azonban a hulladék magában foglalja valamilyen értékes dolog túlzott vagy nem hatékony felhasználás miatti, illetve nem hatékony vagy nem megfelelő módon történő felhasználás miatti károsodását vagy veszteségét is. Ez a tágabb értelmezés döntő fontosságú, amikor a "digitális hulladékot" a digitális lean gyártás területén vizsgáljuk, mivel hangsúlyozza, hogy a digitális hulladék két különböző kategóriáját kell elismerni:

- Passzív digitális hulladék, amely abból ered, hogy nem használják ki a rendelkezésre álló adatokat, és ezáltal lemondanak a bennük rejlő lehetőségeik kiaknázásáról.
- Aktív digitális pazarlás, amely az adatbőségben gazdag gyártási környezetekben keletkezik, ahol nem megfelelő az információkezelési gyakorlat, ami a döntéshozatali folyamatok számára nyújtott információk időzítése, mennyisége vagy relevanciája tekintetében eltérésekhez vezet [5].

3.3 A kiber-fizikai termelési rendszerekre (CPPS) való áttérés: A termelés jövője

A gyártóipar jelentős átalakuláson megy keresztül a kiber-fizikai termelési rendszerek (CPPS) megjelenésével. Ezek a rendszerek paradigmaváltást jelentenek a gyártási folyamatok irányításában és végrehajtásában, mivel a fizikai és a digitális világot eddig soha nem látott módon ötvözik. A CPPS-ek úttörő fejlődést jelentenek a gyártási rendszerekben, ahol a fizikai és a digitális világ közötti határok elmosódnak. Ezek a rendszerek szinergikusan kombinálják a hardvert - beleértve a gépeket, robotokat és érzékelőket - olyan

kifinomult szoftverekkel, mint az adatelemzés és a gépi tanulási algoritmusok. Ez az integráció dinamikus kölcsönhatást tesz lehetővé a gyártás fizikai összetevői és az azokat irányító digitális intelligencia között. Az eredmény egy nagymértékben adaptív és önoptimalizált termelési környezet, amely dinamikusan képes reagálni a változó igényekre és körülményekre [6].

A kiber-fizikai termelési rendszerek kritikus eleme, a digitális iker fogalma. Ez a technológia a fizikai gyártási folyamat digitális másolataként szolgál, lehetővé téve a valós idejű nyomon követést és elemzést. A digitális iker megkönnyíti a fizikai és a digitális világ közötti közel valós idejű visszacsatolási folyamatot, lehetővé téve a gyártási eljárások gyors kiigazítását és optimalizálását. Ezeknek a rendszereknek a lényege a felhőalapú megoldások és a valós idejű adatelemzés kiaknázásának képessége, amely a termelés minden egyes aspektusába részletes betekintést nyújt. Ez a fokozott átláthatóság megváltoztatja a helyzetet, és nélkülözhetetlennek bizonyul a teljesítményhiányok feltárásához, a karbantartási igények előrejelzéséhez és a gyártórendszerek optimális működésének biztosításához. Ez az út azonban nem mentes a kihívásoktól, különösen a kis- és középvállalkozások számára. Az átmenet szükségessé teszi az alternatív módszerek megfontolását, amelyek megvalósíthatóak ezen vállalatok számára. Fontos megemlíteni a multimodális adatgyűjtés, a felhőalapú feldolgozás és a központosított elemzőrendszer kritikusságát. Ezek az összetevők létfontosságúak a kis- és középvállalkozások számára ahhoz, hogy a CPPS-ek előnyeit az új informatikai infrastruktúrába való jelentős beruházás nélkül ki tudják használni [6].

A CPPS-ek felé való elmozdulás több mint technológiai fejlesztés; ez jelentős előrelépést jelent a gyártástechnológiában. Ezek a rendszerek nem pusztán a meglévő folyamatok továbbfejlesztését jelentik, hanem a gyártás digitális korban történő működésének teljes újragondolását. A CPPS-ek integrációja a gyártás hatékonyabb, alkalmazkodóképesebb és intelligensebb jövőjét vetíti előre, amely valóban kiaknázza az ipar 4.0 teljes potenciálját. Ez az átmenet nem csupán a technológiai fejlődéssel való lépéstartásról szól, hanem arról is, hogy új mércét állítsunk fel a gyártás működésére az egyre inkább digitális világban [6].

4 Ipari folyamatok szimulációja

A szimuláció hatékony eszköz az ipari folyamatok megértéséhez, elemzéséhez és optimalizálásához. A szimuláció során egy valós rendszer digitális reprezentációját vagy modelljét hozzák létre, amelyet aztán elemezni lehet, hogy azonosítsuk a hiányosságokat és teszteljük az alternatív stratégiákat. Az ipari folyamatszimuláció segíthet a szervezeteknek a működésük javításában, a költségek csökkentésében és az általános teljesítmény fokozásában [7].

A szimuláció jelentősége az ipari folyamatokban:

Komplex rendszerek megértése

Az ipari folyamatok gyakran számos egymással összekapcsolt komponenst és változót tartalmaznak, amelyeket nehéz lehet megérteni és kezelni. A szimuláció segít az összetett rendszereket kezelhető modellekre bontani, lehetővé téve a mérnökök és a vezetők számára, hogy jobban megértsék, hogyan hatnak egymásra az egyes komponensek és hogyan befolyásolják a folyamat egészét [7].

A szűk keresztmetszetek és a hatékonysági hiányosságok azonosítása

A különböző forgatókönyvek és működési feltételek szimulálásával a szervezetek azonosíthatják a szűk keresztmetszeteket, a hatékonysági hiányosságokat és a folyamatokon belül javítandó területeket. Ez megalapozottabb döntéshozatalt és célzott fejlesztési erőfeszítéseket eredményezhet [7].

Alternatív stratégiák tesztelése

A szimuláció lehetővé teszi a szervezetek számára, hogy különböző folyamat-konfigurációkat, működési stratégiákat és termelési ütemterveket teszteljenek a valós működés megzavarása nélkül. Ez segíthet a leghatékonyabb megközelítések azonosításában és a változások végrehajtásával kapcsolatos kockázatok minimalizálásában [7].

A költségek csökkentése és a teljesítmény javítása

Az ipari folyamatszimuláció segíthet a vállalatoknak optimalizálni működésüket, ami költségcsökkenéshez, jobb áteresztőképességhez, jobb termékminőséghez és nagyobb vevői elégedettséghez vezet [7].

4.1 Szimuláció fajtái

4.1.1 Diszkrét esemény szimuláció

A szimuláció ezen típusa olyan rendszereket modellez, amelyekben az események időben diszkrét időpontokban következnek be. Általában olyan iparágakban használják, mint a gyártás, a logisztika és a szállítás, hogy elemezzék és optimalizálják a sorban állást, erőforrás-elosztást és ütemezést magában foglaló folyamatokat [8].

A diszkrét esemény szimuláció legfontosabb szempontjai és alkalmazásai:

Rendszerelemek:

A modell jellemzően egységekből, eseményekből és erőforrásokból állnak. Az egységek a rendszerben mozgó objektumok (pl. termékek egy gyártósoron), míg az események azokat a jelentős időpontokat jelentik, amikor változások következnek be (pl. gépmeghibásodás, új megrendelések érkezése). Az erőforrások az egységek feldolgozásához szükséges elemek, például gépek, operátorok vagy járművek. [8]

Esemény vezérelt megközelítés:

A diszkrét eseményalapú szimuláció az események bekövetkezésére és a rendszerre gyakorolt hatásukra összpontosít. A szimuláció időrendi sorrendben halad, a modell csak akkor frissül, amikor egy releváns esemény bekövetkezik. Ez az esemény vezérelt megközelítés lehetővé teszi az összetett rendszerek hatékony ábrázolását, mivel nem igényel folyamatos frissítéseket vagy számításokat [8].

Sorba állítás és erőforrás-elosztás:

A diszkrét eseményszimuláció egyik elsődleges alkalmazási területe a sorbanállási rendszerek és az erőforrás-elosztás elemzése. Segítségével meghatározhatók az optimális erőforrás-kihasználási stratégiák, például a várakozási idő minimalizálásához szükséges gépek vagy operátorok száma, és értékelhető a különböző sorbanállási irányelvek (pl. érkezési sorrendben történő kiszolgálás) hatása a rendszer általános teljesítményére [8].

Ütemezés és tervezés:

Ez a szimuláció hasznos az ütemezési és tervezési folyamatok, például a termelési ütemtervek, szállítási útvonalak és karbantartási tervek elemzésére és optimalizálására is. A különböző forgatókönyvek és korlátozások szimulálásával a döntéshozók azonosíthatják a

lehetséges szűk keresztmetszeteket, értékelhetik a különböző alternatívák közötti kompromisszumokat, és hatékonyabb stratégiákat hajthatnak végre [8].

Teljesítményértékelés és optimalizálás:

A diszkrét eseményszimuláció értékes betekintést nyújthat egy rendszer teljesítményébe különböző körülmények között, például a kereslet, az erőforrások rendelkezésre állásának vagy a feldolgozási időnek a változása esetén. A javításra szoruló területek azonosításával és a különböző megoldások tesztelésével a szervezetek optimalizálhatják folyamataikat és megalapozottabb döntéseket hozhatnak [8].

Szoftvereszközök:

Számos szoftvereszköz áll rendelkezésre, az általános célú programozási nyelvektől (pl. Python, Java) a szimulációs könyvtárakkal együtt a speciális szimulációs szoftverekig (pl. Visual Components, Factory I/O). Ezek az eszközök megkönnyítik a diszkrét esemény szimulációs modellek fejlesztését, végrehajtását és elemzését, lehetővé téve a felhasználók számára, hogy rendszereiket intuitívabb módon vizualizálják és kölcsönhatásba lépjenek velük [8, 9].

4.1.2 Folyamatos szimuláció

A folyamatos szimuláció olyan rendszerek modellezésére szolgáló megközelítés, ahol a változók folyamatosan változnak az idő múlásával, szemben a diszkrét esemény szimulációval, amely a különálló, diszkrét eseményekkel rendelkező rendszerekre összpontosít. A folyamatos szimulációt általában olyan iparágakban alkalmazzák, mint a vegyipari feldolgozás, a folyadékdinamika és a környezetmérnöki tervezés, ahol a folyamatok folyamatos áramlásokat és kölcsönhatásokat tartalmaznak [7].

A folyamatos szimuláció legfontosabb szempontjai és alkalmazásai:

Differenciálegyenletek:

A modell használata közben gyakran differenciálegyenletekre támaszkodnak a változók közötti kapcsolatok és azok időbeli változásainak leírására. Ezek az egyenletek különböző technikákkal, például az Euler-módszerrel vagy a Runge-Kutta-módszerrel numerikusan megoldhatók, hogy a rendszer viselkedésének időbeli sorozatait kapjuk [10].

Rendszerdinamika:

A folyamatos szimuláció egyik népszerű megközelítése a rendszerdinamika, amelyet Jay Forrester dolgozott ki az 1950-es években. A rendszerdinamikai modellek készletekből (erőforrások vagy anyagok felhalmozódása) és áramlásokból (változás mértéke) állnak, amelyeket visszacsatolási hurkok kötnék össze. Ezeket a modelleket komplex, dinamikus rendszerek elemzésére és a különböző politikák és stratégiák hosszú távú hatásainak feltárására használják [7].

Számítógépes áramlástan ("Computational fluid dynamics", CFD):

A folyamatos szimulációt gyakran használják a folyadékdinamika területén a folyadékok különböző körülmények közötti viselkedésének tanulmányozására. A CFD modellek részleges differenciálegyenletek, például a Navier-Stokes-egyenletek megoldását foglalják magukban a folyadékáramlás, a hőátadás és más kapcsolódó jelenségek előrejelzésére [11].

5 Folyamatszimulációk alkalmazása az iparban

A folyamatszimulációk létfontosságú szerepet játszanak a modern gyártásban, lehetővé téve a mérnökök számára a gyártási folyamatok tervezését, tesztelését és optimalizálását, mielőtt azokat a való világban megvalósítanák. Ez költségmegtakarítást, jobb termékminőséget, rövidebb gyártási időt és nagyobb működési hatékonyságot eredményez [12].

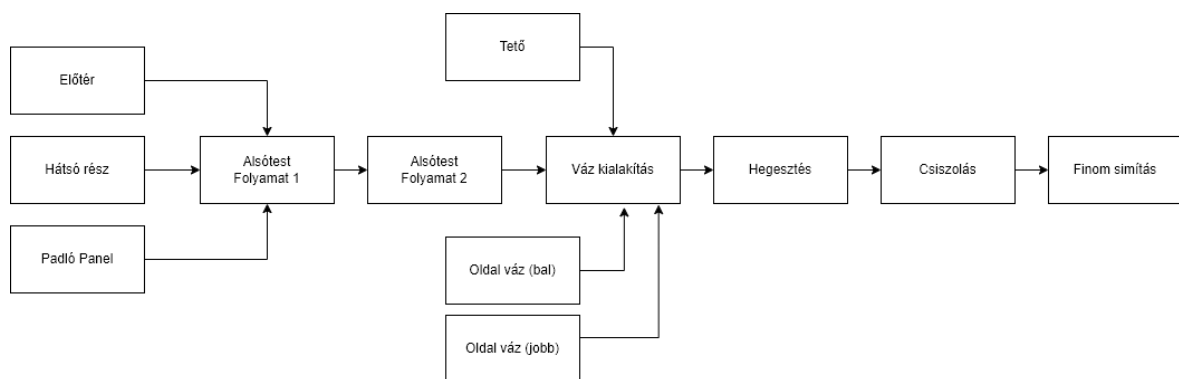
5.1 Esettanulmány

5.1.1 Autóipar (karosszéria gyártás)

Az autóipar kiváló példa a folyamatszimulációk hatékony használatára. Az új autómodellek fejlesztése során például szimulációkat alkalmaznak különböző szempontok, például az aerodinamika, a biztonság és az üzemanyag-hatékonyság tesztelésére, mielőtt fizikai prototípusok készülnének [13].

Egy autógyár karosszériagyárában az összeszerelési szakaszban készül a karosszéria, amely előre megformázott fémdarabokból áll. Ez a bonyolult folyamat több mint száz hegesztőrobotot és különféle egyéb berendezéseket igényelhet a karosszéria összeállításához, mielőtt az a következő gyártási fázisba kerülne. Mivel a karosszériák jelentős eltéréseket mutatnak az egyes új autómodellek esetében, a gyártók gyakran szembesülnek azzal, hogy szinte minden új modellhez egy teljesen új karosszériaüzemet kell tervezniük. Ez a feladat egyre nagyobb kihívást jelent, mivel az autóiparban a termékek életciklusa egyre rövidül, és az új berendezésekhez szükséges beruházás gyakran meghaladja

a százmillió dollárt. A kezdeti koncepcionális tervezési fázis, a tervezési folyamat egyik legkorábbi lépése, a műhely 12-18 különböző blokkra való felosztását jelenti. Minden egyes blokk egy hegesztési területet jelent, amely számos hegesztési műveletnek ad helyet a különböző állomásokon. A gyártási folyamat szétválasztása érdekében a két egymást követő blokk közé jellemzően puffereket építenek be, ami egy blokk-puffer struktúrát eredményez. Ez a struktúra egy átalakuló topológiát mutat, ahol az egyes blokkokból származó részegységek és alegységek a következő területeken konvergálnak. Például az olyan alkatrészek, mint a gépkocsi alváza, a tető és az oldalsó keretek a keretező sornak nevezett részen kerülnek összeszerelésre. Az ebben az esettanulmányban figyelembe vett karosszériaépítési struktúrát az 1. ábra mutatja [14].



2. ábra: Egy karosszéria folyamat koncepcióterve [14]

A karosszéria gyártása az autóiparban nagyrészt egységes sémát követ, a fő gyártási lépésekben csak kisebb eltérések vannak a különböző gyártók és autómodellek között. A karosszériaüzem koncepciója jellemzően meghatározott számú (m) blokkot foglal magában, ahol mind az emberi munka, mind a robotizált folyamatok szerveződnek. Konkrét példánkban (lásd a 2. ábrát) a gyártás a karosszériaelemek elkészítésével kezdődik az adott blokkokban, 3 részegység bevonásával. Miután ezek az elemek elkészültek, a következő két blokkban egyesítik és összehegesztik őket. Ezt követően az oldalkereteket és a tetőt az alvázhoz rögzítik, ami a karosszériaváz kialakítását eredményezi. Ezt követően a karosszéria csiszolási tevékenységeken megy keresztül. A finom simítás, befejező részlegben különböző alkatrészek, például az ajtók, a csomagterfedél és a hátsó ajtófélfák rögzítése történik. Ezután, a kész karosszéria elhagyja a karosszériaüzemet, és a festőműhelybe szállítják. A gyártási folyamat egyik fő kihívása a blokkokon belüli véletlenszerű meghibásodások, amelyeket gyakran a robotok meghibásodása okoz. Ezeknek a meghibásodásoknak az időtartama a valós körülmények között jelentősen változik. Az egyes blokkok (s) megbízhatóságát és

rendelkezésre állását, ahol $s \in \{1, \dots, m\}$, két kulcsfontosságú mérőszám határozza meg: a javításig eltelt átlagos idő (MTTR) és a meghibásodások közötti átlagos idő (MTBF) [15].

$$Elérhetőség_s = \frac{MTBF_s}{MTBF_s + MTTR_s}$$

Egy magasan automatizált rendszerben, mint például egy karosszériaüzemben, feltételezhetnénk, hogy a ciklusidő állandó. Ezek azonban blokkról blokkra változhatnak. Fontos megjegyezni, hogy ebben az összefüggésben a "ciklusidő" kifejezés az egy blokkban végzett műveletsorozat befejezéséhez szükséges időre utal, nem pedig a teljes gyártási folyamat időtartamára (teljes ciklusidő). Ami a teljes ciklusidőt illeti, az optimalizálás nem megvalósítható, mivel a tevékenységeket a műszaki korlátok miatt nem lehet más blokkszerkezetbe átszervezni [14].

Egy adott blokk (s) maximális elméleti ciklusideje T_s^{max} a blokk rendelkezésre állásának és az üzem szükséges gyártási sebességének figyelembevételével határozható meg. Itt a rendelkezésre állás a blokk üzemidejére vagy működési idejére utal, míg a termelési ráta arra az egység számra, amelyet egységnyi idő alatt kell előállítani a szükséglet kielégítéséhez [14].

$$T_s^{max} = \frac{GyártásiIdő * Elérhetőség_s}{TermelésiRáta}$$

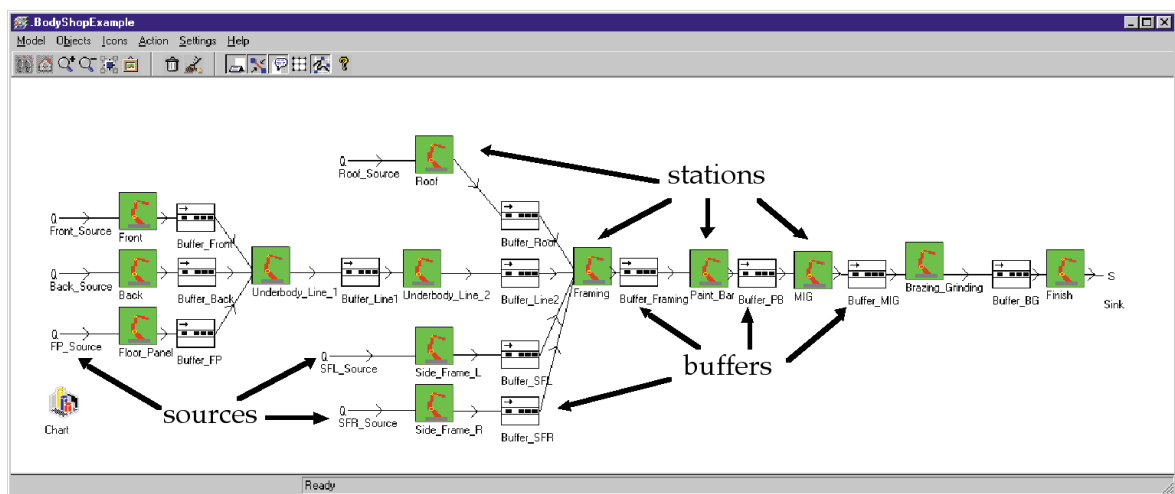
A koncepcionális tervezési fázis másik célja, hogy minden egyes blokk számára olyan ciklusidőt határozzon meg, amely szorosan illeszkedik az elméleti maximális ciklusidőhöz. Ezen felső határértékhez közeli ciklusidő elérése a lehető legnagyobb időtartamot kínálja a blokkon belüli munka megszervezéséhez a következő tervezési fázisban. Ez viszont viszonylag nagyfokú rugalmasságot biztosít a mérnökök számára a blokk(ok) részletes tervezésénél. Ez még a szükséges berendezések mennyiségének és következésképpen a szükséges beruházásoknak a csökkentését is elősegítheti, mivel biztosítja, hogy minden tevékenységet a ciklusidőn belül végezzenek el [14].

Tekintsük a következő példát: ha a karosszériaüzemet úgy tervezték meg, hogy napi 400 darabot gyártson 1000 perc gyártási időn belül, és az elérhetőség 90%-os, akkor a keretelési terület maximális ciklusideje

$$T_{Karosszéria}^{max} = \frac{1000 * 60 * 0,9}{400} = 135$$

azaz 135 másodpercben lenne kiszámítható [14].

Egy lehetséges szimulációs megoldás a SIMPLE++, ami egy szimulációs eszköz, amelyet a napi kimenet kiszámítására alkalmas szimulációs modell létrehozására használnak. Különösen előnyös, mert lehetővé teszi egy olyan animációs elrendezés létrehozását, amely igazodik a gyártási folyamat blokkjainak és pufferjeinek szerkezetéhez. A SIMPLE++ segítségével szimulációs kísérletsorozatot végeztek lépésenként, ami a karosszériaépítési probléma megoldásának általános megközelítése. A forrás azonban elismeri, hogy ez a módszer gyakran lokális optimumot eredményez - nem feltétlenül a legjobb átfogó megoldást [14].

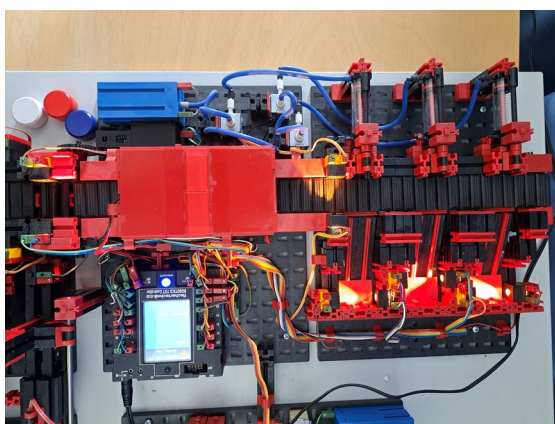


3. ábra: Simple++ szimulációs modell egy karosszériaüzlet koncepciójáról [14]

6 Anyag és módszertan

Dolgozatom fő célkitűzéseinek megvalósítása érdekében először is el kellett döntenem, hogy milyen eszközöket fogok felhasználni a projekt megvalósításához. Az ötlet maga gyártócellák órák oktatása közben született, melyek során megismerkedtem a különböző folyamatok felépítésével és logikájával.

Az oktatásban számos fizikai, egyszerű példákat bemutató, kompakt, szenzorokkal és végrehajtókkal teli állomást találunk (Festo válogató, adagoló, kupakoló állomások; Fischertechnik színválogató állomás stb.)



4. ábra: Fischertechnik válogató állomás (saját fotó)

Ezen eszközök vezérlése lényegében digitális logikán alapszik melynek megértése nem mindig egyértelmű. Feladat kitűzésem az lett, hogy valamilyen módon elemezni tudjam az elem tárt folyamatokat. Mindenképpen egy szimulációs környezetet is ki szerettem volna alakítani, hiszen az bármikor velem lehet laptopon, nem kell a fizikai eszközöktől függeni. Illetve számos fizikai bekötés hibaelhárítás is elkerülhető, ha digitális eszközmegoldást alkalmazok.

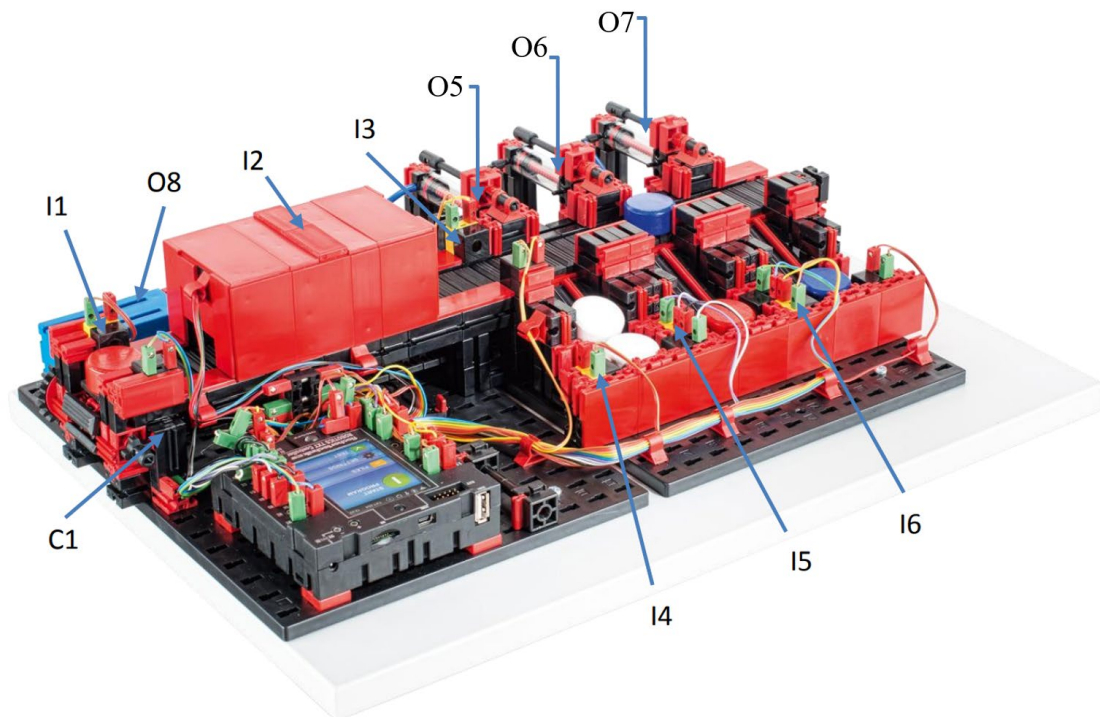
Az 5. ábrán látható állomások különböző vezérlőkkel is irányíthatóak. A lényeg, hogy digitális/analóg be- és kimenetek felhasználásával történjen a vezérlés. A meglévő folyamaton kívül egy 1:1 szimulációs megoldást is elkészítettem, így könnyebben és hatékonyabban tudtam a változtatásokat kipróbálni a folyamaton a fizikai modellen való változtatások előtt. A következő fejezetekben bemutatom, hogy milyen eszközöket használtam fel egy ipari folyamat elemzéséhez.

6.1 Felhasznált eszközök

6.1.1 Fischertechnik színválogató folyamat modell

A Fischertechnik egy sokoldalú, moduláris építőrendszer, amelyet eredetileg oktatási célokra fejlesztettek ki. Széles körben használják az oktatásban és a kutatásban valós ipari folyamatok modellezésére és szimulálására, köszönhetően alkalmazkodóképességének és az összetevők széles választékának. A projektem keretében a Fischertechnik gyakorlatias és hatékony eszközt biztosított egy színválogató folyamat fizikai modelljének megépítéséhez, ami elengedhetetlen az automatizált gyártási és válogatórendszerek alapelveinek megértéséhez és bemutatásához. A színérzékeléssel ellátott válogatósor a különböző színű építőelemek automatikus szétválasztására szolgál. Ebben a folyamatban egy szállítószalag geometriailag azonos, de különböző színű építőelemeket szállít egy színérzékelőhöz, ahol színük szerint történik a szétválasztás. A szállítószalagot egy egyenáramú motor hajtja, a szállítási útvonalat pedig egy impulzuskapcsoló segítségével érzékeli. A munkadarabok kidobását pneumatikus hengerek végzik, amelyek a megfelelő tárolóhelyekhez vannak rendelve, és amelyeket mágnesszelepek működtetnek. Több fénysorompó figyel a munkadarabok áramlását és a tárolóhelyek töltöttségi szintjét. A színérzékelést egy optikai színérzékelő végzi, amely a felületi visszaverődés alapján képes érzékelni a színeket. Technikailag tehát a színérzékelő egy reflektív érzékelő, amely azt jelzi, hogy egy felület mennyire jól veri vissza a fényt. Az érzékelő által mért érték ezért nem arányos a mért szín hullámhosszával, és még a színekoordináták vagy színterek (pl. RGB vagy CMYK) hozzárendelése sem lehetséges. A tárgy színe mellett a környezeti fény, a tárgy felülete és a tárgynak az érzékelőtől való távolsága is befolyásolja a visszaverődés minőségét. Emiatt elengedhetetlen, hogy a színérzékelő védve legyen a környezeti fénytől, és a tárgyak felülete hasonló legyen. Ezenkívül fontos, hogy az érzékelőt a tárgy felületére merőlegesen szereljék fel. Az egyes színek mért értékeit korlátozó küszöbértékek különbséget tesznek a színes munkadarabok között. Az adagolást az első kidobó szerkezet előtt elhelyezett fénysorompó segítségével lehet szabályozni. Az érzékelt színértéktől függően a megfelelő pneumatikus henger késleltetve lép működésbe, miután a munkadarab belépett a fénysorompó érzékelő terébe. Ezen a ponton az impulzuskapcsoló kerül használatra, amely érzékeli a szállítószalagot hajtó fogaskerék forgását. Az időfüggő késleltetéssel ellentétben ez a megközelítés ellenáll a szállítószalag sebességében bekövetkező zavaroknak. A kidobott munkadarabok három csúszdán keresztül jutnak el az egyes tárolóhelyekre. A tárolóhelyek fénysorompókkal vannak felszerelve, amelyek érzékelik, hogy a tárolóhely tele van-e vagy

sem. A fényzorompók azonban nem tudják megmondani, hogy hány munkadarab van a tárolóhelyen [16].



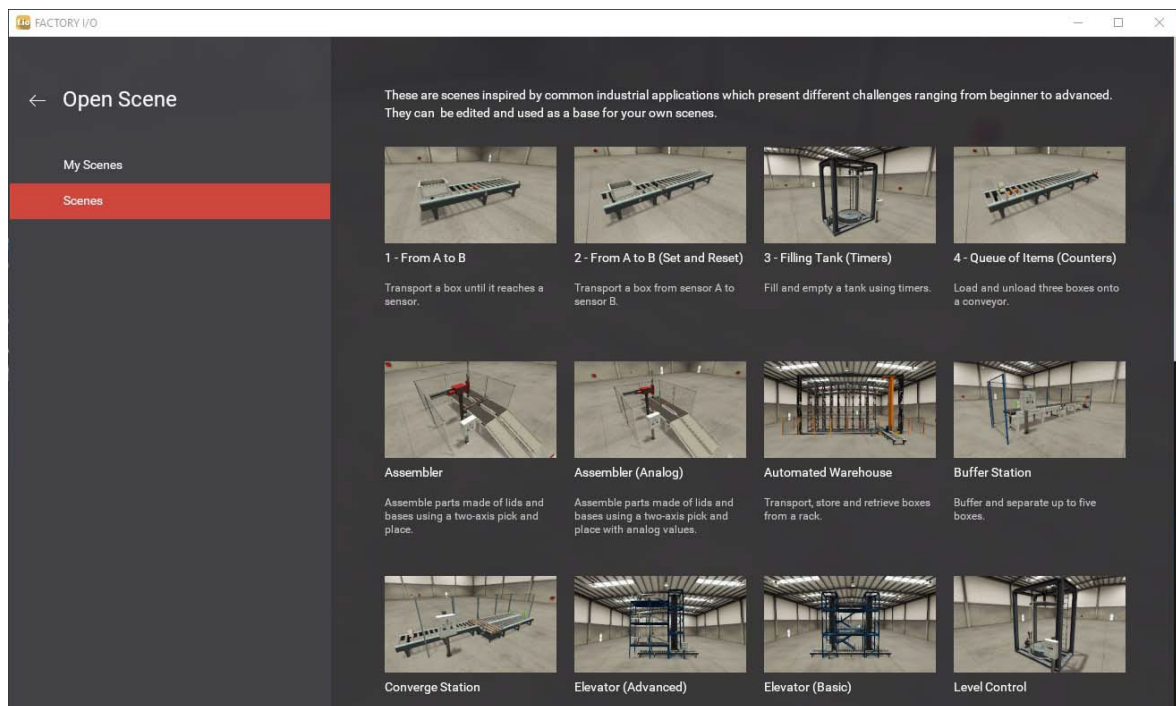
5. ábra: Fischertechnik színválogató állomás [16]

2. táblázat: Szenzorok és aktuátorok [16]

Fototranzisztor	I1, I3, I4 I5, I6
Színérzékelő	I2
Impulzus gomb	C1
Szállítószalag motor	M1
Szelep kidobó fehér	O5
Szelep kidobó piros	O6
Szelep kidobó kék	O7
Kompresszor	O8

6.1.2 Factory I/O

A Factory I/O egy 3D-s gyári szimulációs program, amelyet az ipari automatizálásban való képességek fejlesztésére terveztek. A platform legfőbb előnye a sokoldalúságában rejlik, amely lehetővé teszi a felhasználók számára, hogy számos PLC-vel, vagy akár az integrált soft PLC-vel szimulálják a folyamatokat. Sokoldalú szimulációs képességek. A Factory I/O szabványos ipari alkatrészek átfogó könyvtárával rendelkezik. Ez biztosítja, hogy a felhasználók bonyolult virtuális gyárakat tervezhessenek, valós forgatókönyveket reprodukálva. A Factory I/O leegyszerűsíti a folyamatokat. Előre elkészített jelenetekkel rendelkezik, amelyek az elterjedt ipari automatizálási alkalmazásokat tükrözik. Ez a funkció biztosítja, hogy a felhasználók a kezdeti beállítási gondok nélkül azonnal belevethetik magukat a PLC-programozással kapcsolatos projektekbe [17].

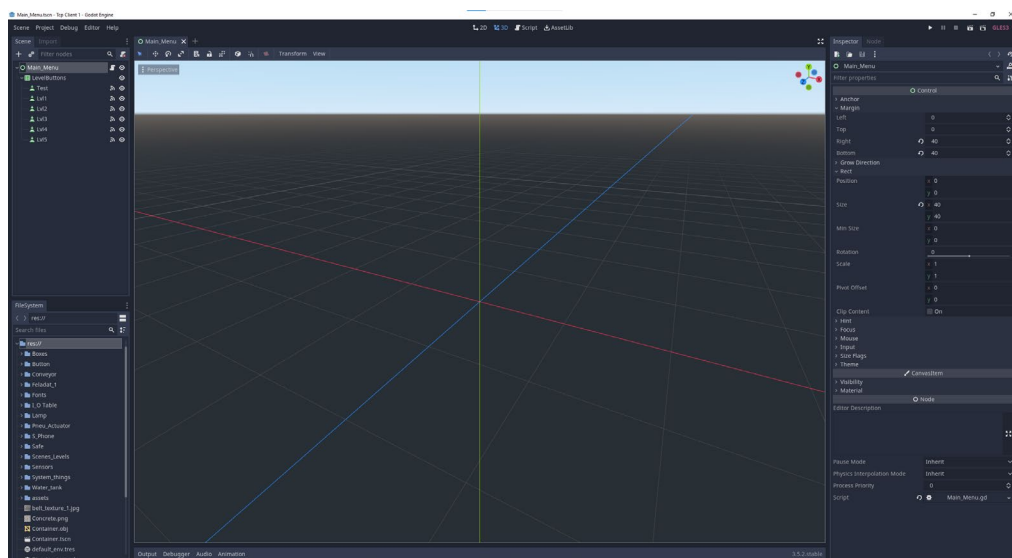


6. ábra: Factory I/O jelenetek [17]

6.1.3 Godot

A Godot egy olyan játékmotor, amely számos szimulációs funkciót kínál, így kiváló választás a projektemhez. Néhány indok, ami miatt ezt választottam:

- Nyílt forráskódú és ingyenes
- Könnyen programozható
- Beépített hálózati funkciók a szimulációs kommunikáláshoz
- Nagymértékű testreszabhatóság
- 3D (és 2D) vizualizáció támogatása
- Cross-Platform (több eszközön is használható)

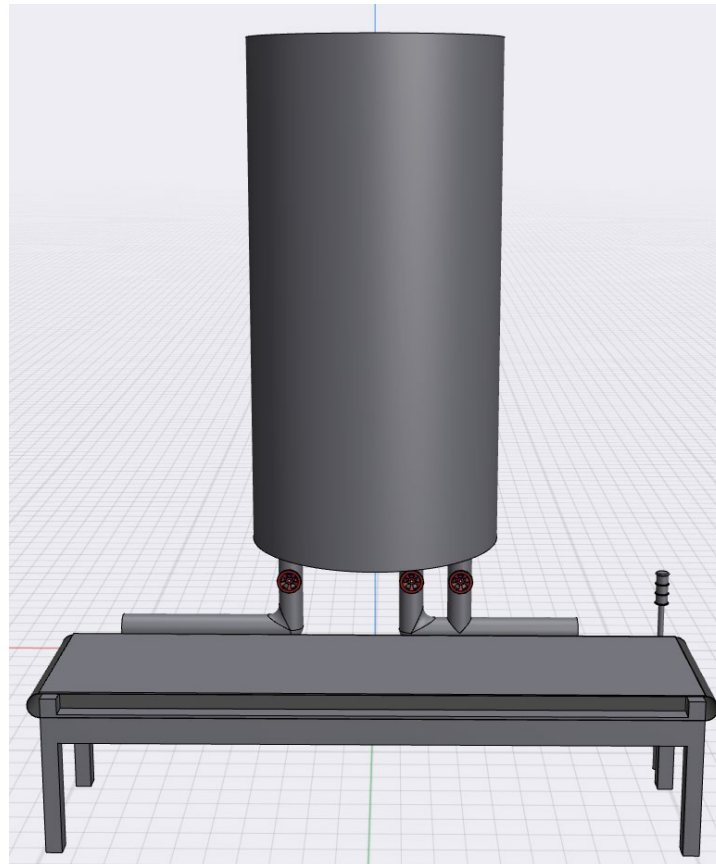


7. ábra: Godot fejlesztőkörnyezet (képernyő fotó)

6.1.4 Shapr3D

A projekthez szükséges 3D-s modellek Shapr3D-s CAD szoftverben készítettem el. A Shapr3D egy innovatív 3D modellező eszköz ipari tervezők, mérnökök, termékmenedzserek, gyártók és más tervező szakemberek számára. A Shapr3D küldetése, hogy megreformálja a múltban megrekedt CAD ipart. Az Apple Pencil kompatibilis iPadeken teljes közvetlen modellezési funkciókészletet biztosít. A gyártói minőségű 3D modellező eszköz Siemens™ Parasolid™ geometriai modellező magot és D-Cubed™ sketch engine-t használ. Kompatibilis az összes jelentős asztali CAD szoftverrel, és lehetővé teszi a felhasználók számára az X_T, STEP, IGES, STL, DXF, DWG, JPG, PNG fájlok közvetlen importálását, valamint az X_T, STEP, DWG, DXF, STL, OBJ fájlok exportálását. Az alkalmazást eredetileg az OpenCASCADE kernel használatával fejlesztették ki. 2017-

ben a csapat partnerségre lépett a Siemens PLM-mel, hogy integrálja a Siemens™ Parasolid™ geometriai modellező kernelt. A Shapr3D az első olyan CAD-rendszer, amely natívan futtatja a Parasolid™ szoftvert az iPad-en. A Shapr3D-nek kereskedelmi és egyetemi felhasználói egyaránt vannak. A szoftvert használó oktatási intézmények közé tartozik az MIT, a Harvard, a Stanford, a Columbia, a Cambridge, a University of Tokyo és a Singapore Polytechnic, míg a szakemberek között terméktervezők, mérnökök, építészek és ékszertervezők találhatók [18].

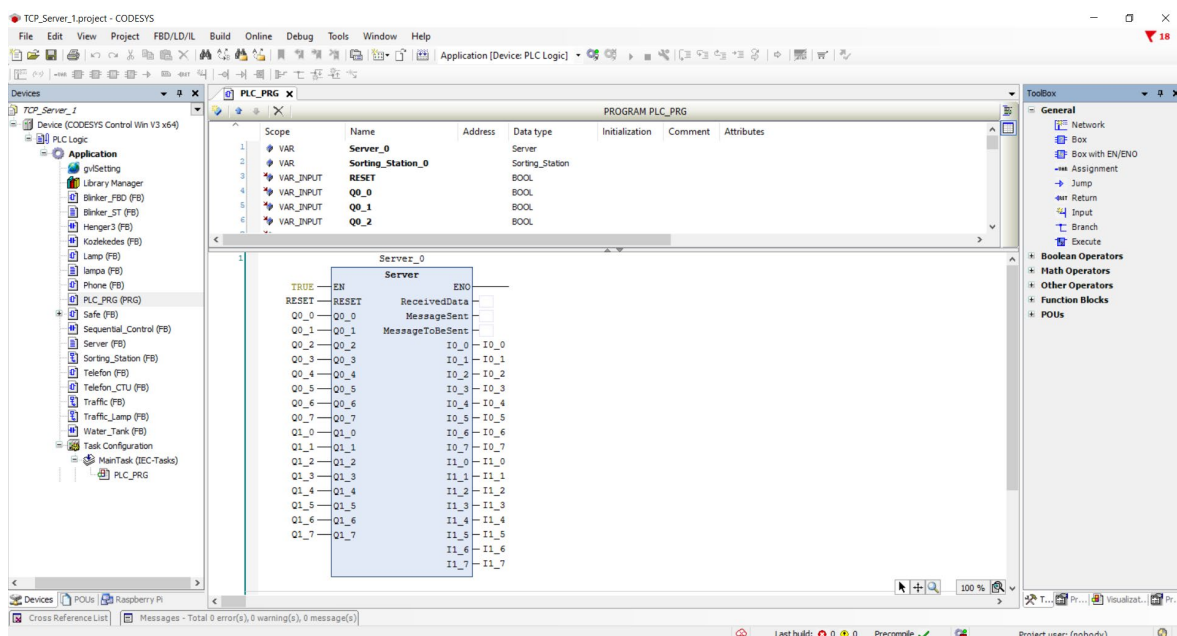


8. ábra: Tartály, futószalag és andon lámpa modelljei (képernyő fotó)

6.1.5 CODESYS

Ahhoz, hogy a szimulációs megoldásom valamilyen logika, vezérlés szerint működjön (kommunikáljon) szükségem volt egy vezérlő rendszere. Így esett a választásom a CoDeSys szoftverre.

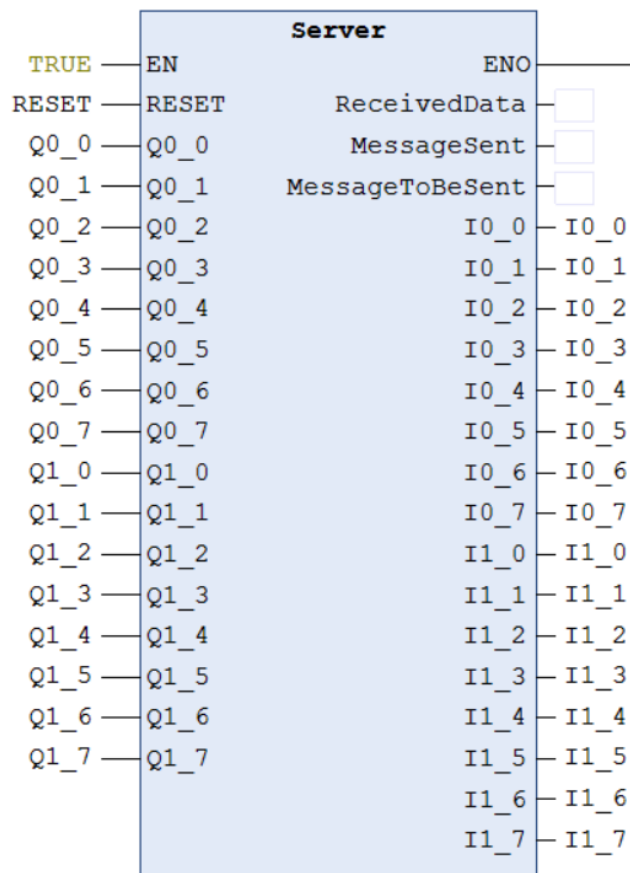
A CODESYS Group által létrehozott CoDeSys egy olyan fejlesztőkörnyezet, amelyet az IEC 61131-3 ipari szabványnak megfelelő PLC-k programozására terveztek. Alapvetően a PLC-gyártók vagy saját szoftver megoldást fejlesztenek, vagy harmadik féltől származó forrást bíznak meg a szoftverük integrálásával. A második esetben az ipari automatizálásban az egyik legáltalánosabb szoftver megoldás a CoDeSys. A Codesys, bármelyik PLC gyártóval használható, és hardverfüggetlen.



9. ábra: CoDeSys fejlesztőkörnyezet (képernyő fotó)

6.1.6 PLC TCP/IP szerver

A CoDeSys 3.5 SP18 Patch 4-es verziójában kezdtem meg a fejlesztést. Eszközként a CODESYS Control Win V3 x64 használtam [19]. Az iparilag széles körben alkalmazott TCP/IP protokollt használtam a kommunikációhoz. A PLC szerver (továbbiakban virtuális PLC) 16db digitális be- és kimenettel rendelkezik. Ezek elnevezései I0_0, I0_1, Q0_0, Q0_1, stb. Az I/O-k elnevezéseiben az első szám jelzi, hogy hányadik byte míg a második szám jelzi, hogy hányadik bit memóriában tároljuk el a digitális jelet. Így például az I0_0 jelét a 0.byte 0.bitjén tárolom.



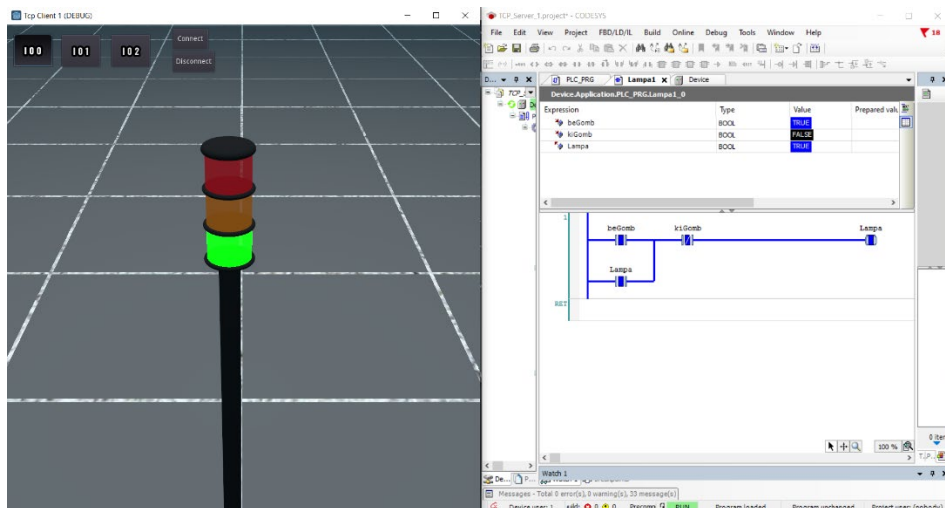
10. ábra: Virtuális PLC (képernyő fotó)

A jeleket a már említett TCP/IP protokoll szerint továbbítottam, nem pedig fizikális vezetékeken, mint ahogy ez történne az I/O protokoll szerint. A virtuális PLC egy string üzenetet küld a kliens részére, majd várja a kliens üzenetét. Az üzenetek felépítése a következő:



11. ábra: String üzenet felépítése (saját ábra)

A kapott üzenetet a receiver fél feldolgozza és a megfelelő üzenetcsomagot küldi vissza. Például, ha a folyamatszimuláció UI felületén lenyomunk egy gombot, ami az I0_0 virtuális bementre van kötve akkor a kliens üzenete a következő lesz amíg a gombot nyomva tartjuk: #010000000#100000000. Amint látható a 0. byte 0.bitje a logikai 1-es értéket veszi fel. Ezt az üzenetet feldolgozza a virtuális PLC és a rá írt logika alapján visszaküld egy hasonló üzenetet. Például kapcsoljunk fel egy lámpát, ami a Q0_0 virtuális kimentre van kötve. Akkor, ha lenyomjuk a gombot világítson a lámpa. A server üzenete a következő lesz hasonlóan az előző üzenethez: #010000000#100000000.



12. ábra: Program működés közben (saját ábra)

```

Server
1 fbTcpServer(xEnable := Enable, ipAddress := gvlSetting.gc_stIpAddr, uiPort := gvlSetting.gc_uiPort);
2 fbTcpConnection(xEnable := fbTcpServer.xBusy, hServer := fbTcpServer.hServer);
3
4 fbTcpRead(xEnable := fbTcpConnection.xActive, hConnection := fbTcpConnection.hConnection, szSize := sizeof(ReceivedData), pData := ADR(ReceivedData), szCount => udiRead);
5
6 // Detect if client has disconnected
7 IF F_TRIG_Connection.Q THEN
8   Enable := FALSE;
9 END_IF
10
11 IF NOT Enable AND NOT fbTcpConnection.xActive THEN
12   resetTimer(IN := NOT resetTimer.Q, PT := T#100MS); // Start a timer for 100 m seconds
13   IF resetTimer.Q THEN
14     Enable := TRUE;
15     Enable := TRUE;
16   END_IF
17 END_IF
18
19 IF fbTcpRead.xError THEN
20   xRead := FALSE;
21 ELSEIF fbTcpRead.xReady THEN
22   IF LEN(ReceivedData) >= 20 AND MID(ReceivedData, 2, 1) = '#0' AND MID(ReceivedData, 2, 11) = '#1' THEN
23     bit_string_0 := MID(ReceivedData, 8, 3);
24     bit_string_1 := MID(ReceivedData, 8, 13);
25
26     // for I0 variables
27     I0_0 := BYTE_TO_BOOL((bit_string_0[0] - 48));
28     I0_1 := BYTE_TO_BOOL((bit_string_0[1] - 48));
29     I0_2 := BYTE_TO_BOOL((bit_string_0[2] - 48));
30     I0_3 := BYTE_TO_BOOL((bit_string_0[3] - 48));
31     I0_4 := BYTE_TO_BOOL((bit_string_0[4] - 48));
32     I0_5 := BYTE_TO_BOOL((bit_string_0[5] - 48));
33     I0_6 := BYTE_TO_BOOL((bit_string_0[6] - 48));
34     I0_7 := BYTE_TO_BOOL((bit_string_0[7] - 48));
35
36     // for I1 variables
37     I1_0 := BYTE_TO_BOOL((bit_string_1[0] - 48));
38     I1_1 := BYTE_TO_BOOL((bit_string_1[1] - 48));
39     I1_2 := BYTE_TO_BOOL((bit_string_1[2] - 48));
40     I1_3 := BYTE_TO_BOOL((bit_string_1[3] - 48));
41     I1_4 := BYTE_TO_BOOL((bit_string_1[4] - 48));
42     I1_5 := BYTE_TO_BOOL((bit_string_1[5] - 48));
43     I1_6 := BYTE_TO_BOOL((bit_string_1[6] - 48));
44     I1_7 := BYTE_TO_BOOL((bit_string_1[7] - 48));
45
46   ELSE
47     // handle unexpected format
48   END_IF;
49 END_IF
50
51   DataByte_0 := '';
52   DataByte_1 := '';
53   // DataByte_0
54   DataByte_0 := CONCAT(DataByte_0, INT_TO_STRING(BOOL_TO_INT(Q0_0)));
55   DataByte_0 := CONCAT(DataByte_0, INT_TO_STRING(BOOL_TO_INT(Q0_1)));

```

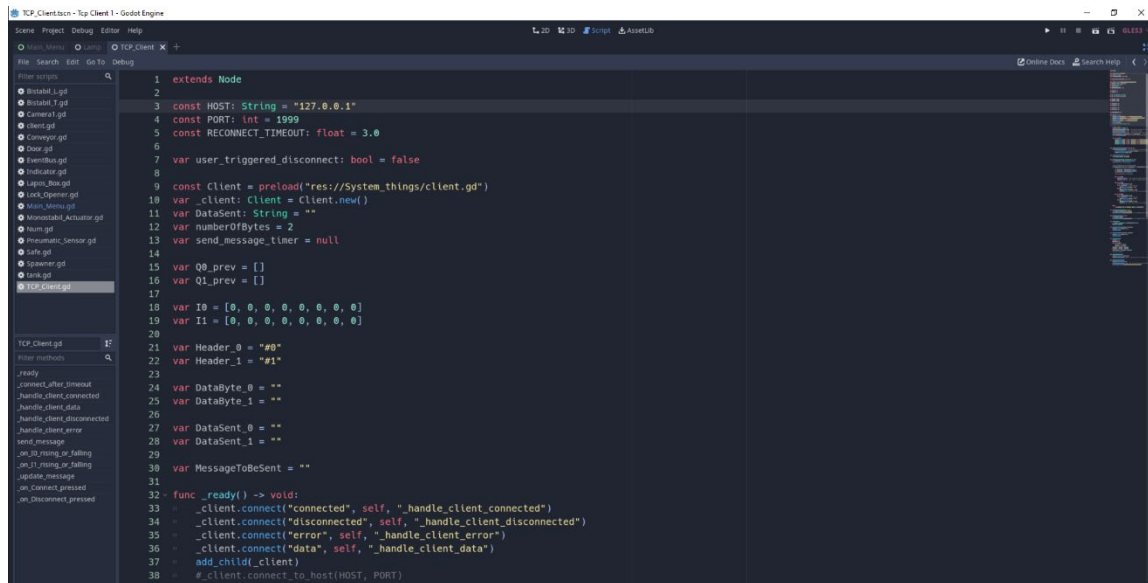
13. ábra: Szerver program részlet/ST (képernyő fotó)

A fenti kód részletben látható miként történik az üzenet feldolgozása és jellé alakítása. A megfelelő bemeneti logikai változót igazzá vagy hamissá teszi a kliens üzenete. Így valósítottam meg a virtuális gombok és érzékelők ”bekötését” a virtuális PLC-be.

Fontos, hogy a fenti konfiguráció lokális hálózaton valósult meg, tehát a két szoftver a 127.0.0.1-es IP címet használta. Viszont a TCP/IP rugalmasságának köszönhetően egy valódi PLC-vel is van lehetőség vezérelni a szimulációs folyamatot. Ez egy továbbfejlesztési lehetőség.

6.1.7 Godot TCP/IP kliens

A Godot egy játék motor, amely nyílt forráskódú, így a vele elkészült programok az alkotó teljes tulajdonát képezik. 2D-s és 3D-s játékok fejlesztésére használatos fejlesztő környezet. Viszont támogatja a TCP/IP kommunikációs protokollt, így egy ipari folyamat szimulációt is lehet vele készíteni. Programozása GDScript nyelven történik mely hasonlít a Python program nyelvhez.



14. ábra: Kliens program részlet/GDScript (képernyő fotó)

A kommunikációhoz egy Eventbus rendszert használtam. A Godot képes belső signalok alapján jelet továbbítani, így nekem csak azt kellett megírni hogy mikor milyen jelet továbbítson.

Az alábbiakban leírom, hogy a jeleknek milyen állapotait vizsgálja a program mind a szerver és kliens oldalon. A már említett virtuális bemenetek és kimenetek állapotait vizsgálva megkülönböztethetünk kétfajtát. Rising edge (felfutó él) illetve falling edge (lefutó él). Amikor a jel 0-ról 1-re vált, tehát alacsony jelről magasra vált, hívjuk Rising edge-nek. Ennek ellentetjét, amikor 1-ről 0-ra vált tehát magas jelből alacsonyra vált, hívjuk Falling edge-nek. Minden változó állapot változására az elküldendő üzenet megfelelő bitje módosul, majd azonnal elküldésre kerül. Valóságban 30ms-es késés van beiktatva az akadások elkerülése végett.

<pre>#0 Byte Inputs signal IO_0_rising signal IO_0_falling signal IO_1_rising signal IO_1_falling</pre>	<pre>#0 Byte Outputs signal Q0_0_rising signal Q0_0_falling signal Q0_1_rising signal Q0_1_falling</pre>
---	--


```
func emit_rising(variable_name):
>| emit_signal(variable_name + "_rising")
>|
func emit_falling(variable_name):
>| emit_signal(variable_name + "_falling")
```

15. ábra: Eventbus rendszer részlet (képernyő fotó)

A különböző gomboknak, szenzoroknak és érzékelőknek a programját az Eventbusra támaszkodva írtam meg. Ennek funkcionális megvalósítását bemutatom az alábbiakban. Egy gombot ha lenyomok akkor a hozzá tartozó rising signal elküldésre kerül melynek hatására az elküldendő üzenet frissül. Az általam elkészített rendszer teljesen automatizáltan működik, kapcsolatot létesít és kommunikál virtuális PLC-vel a megfelelő módon.

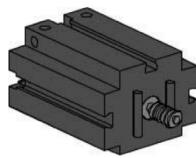
```
1 extends Button
2
3 enum InputEnum {
4   >| IO_0, IO_1, IO_2, IO_3, IO_4, IO_5, IO_6, IO_7,
5   >| I1_0, I1_1, I1_2, I1_3, I1_4, I1_5, I1_6, I1_7
6 }
7
8 var inputEnumNames = [
9   >| "IO_0", "IO_1", "IO_2", "IO_3", "IO_4", "IO_5", "IO_6", "IO_7",
10  >| "I1_0", "I1_1", "I1_2", "I1_3", "I1_4", "I1_5", "I1_6", "I1_7"
11 ]
12
13 export(InputEnum) var selected_input
14
15 func _on_Button_button_down():
16   >| var signal_name_rising = "%s" % inputEnumNames[selected_input]
17   >| EventBus.emit_rising(signal_name_rising)
18
19 func _on_Button_button_up():
20   >| var signal_name_falling = "%s" % inputEnumNames[selected_input]
21   >| EventBus.emit_falling(signal_name_falling)
```

16. ábra: Gomb funkció programja (képernyő fotó)

6.2 Fischertechnik színválogató állomás bemutatása és elemzése

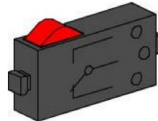
6.2.1 Technikai adatok

S motor: A válogatósor érzékelő funkcióval ellátott szállítószalagját S-motor hajtja. Ez a, állandó mágnessel ellátott egyenáramú motor párosítható egy kiegészítő reduktoros fogaskerékkel. A motor 9 V egyenfeszültséggel működik, és legfeljebb 650 mA áramot vesz fel. Ennek következtében a motor 4,8 mNm csúcsnyomatékot tud produkálni, és üresjáratú fordulatszáma 9500 fordulat/perc. A csatlakoztatott reduktoros sebességváltó 64,8:1 áttételt kínál, és oldalkimenetet biztosít [16].



17. ábra: S motor [16]

Mini-kapcsoló: Ez a nyomógombos kapcsoló egy impulzuskerékkel párosítva inkrementális forgókódolóként működik, és azonosítja a szállítószalag helyzetét. A mini-kapcsoló rendelkezik egy átváltó mechanizmussal, amely lehetővé teszi, hogy normál esetben zárt vagy normál esetben nyitott érintkezőként működjön [16].



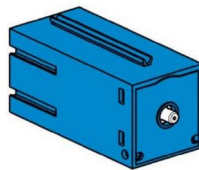
18. ábra: Mini-kapcsoló [16]

Fototranzisztor: A fototranzisztorok fénysorompóként szolgálnak. Ezek a fototranzisztorok egy adott fényerősség elérésekor kapcsolnak, vezetnek az áramot. Ha azonban a fényerő meghalad egy bizonyos küszöbértéket, akkor nem vezetővé válnak. A fototranzisztor egy ráirányított lámpával párosítva vezeti az elektromosságot, és fénysorompóként működik. A külső fény hatásának minimalizálása érdekében kóbor fénytakaró alkalmazható [16].



19. ábra: Fototranzisztor [16]

Kompresszor: A rendszert egy membránpumpa sűrített levegője működteti. Ez a speciális membránszivattyú két kamrából áll, amelyek között egy membrán van. Az egyik kamrában egy bűtyök függőlegesen hajt egy dugattyút, amely vagy beszívja a levegőt, vagy kinyomja azt a szomszédos kamrából. Ahogy a dugattyú leereszkedik, a membrán visszahúzódik, lehetővé téve a levegő belépését a második kamrába a bemeneti szelepen keresztül. Ezzel szemben, amikor a dugattyú emelkedik, a membrán a levegőt a kimeneti szelepen keresztül kinyomja. Ez a kompresszor 9 V szabványos feszültségen működik, és körülbelül 0,7 bar nyomást tud létrehozni. A felvett csúcsáram 200 mA [16].



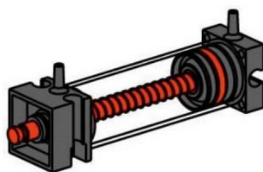
20. ábra: Kompresszor [16]

3/2-es mágnesszelep: A 3/2-utas mágnesszelepek a pneumatikus hengerek irányításának szerves részét képezik. Ezek a szelepek, ahogy a nevük is mutatja, három csatlakozóval és két működési állapottal rendelkeznek. A kapcsolási mechanizmusokat egy mágnestekercs hajtja meg, amely egy rugó ellen dolgozik. Amikor a mágnesszelepre elektromos feszültséget kapcsolunk, a Lorentz-erő hatására a mozgatható henger a rugó ellen hat, ami a szelep kinyílásához vezet. A "nyitva" itt azt jelenti, hogy a sűrített levegő csatlakozónyílása összekapcsolódik a hengernyílással. Amikor a feszültség megszűnik, a rugó visszahelyezi a hengert a hengernyílásba [16].



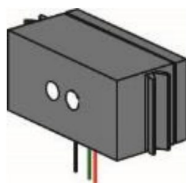
21. ábra: 3/2-es mágnesszelep [16]

Pneumatikus henger: a munkadarabok adagolását három pneumatikus henger végzi. Ezeket a hengereket 3/2-es útváltó mágnesszelepek szabályozzák. Minden egyes pneumatikus hengerben egy dugattyú két rekeszre osztja a belső teret. Az e rekeszek közötti nyomáskülönbség erőt gyakorol a dugattyúra, ami elmozdulásra készíti azt. Ez az elmozdulás a két kamra térfogatának változásához vezet [16].



22. ábra: Pneumatikus henger [16]

Színérzékelő: A színérzékelők jelentős szerepet játszanak az automatizálási technológiában. Alkalmazhatók például az árnyalat vagy a színjelölés ellenőrzésére, így meggyőződhetünk arról, hogy a megfelelő alkatrészek vannak-e összeszerelve. A fischertechnik színérzékelő vörös fényt bocsát ki. A felület színétől függően ez a fény különböző intenzitással verődik vissza. A visszavert fény intenzitását egy fototranzisztor méri, amely aztán egy 0 V és 9 V közötti feszültségértékre fordítja. Érdemes megjegyezni, hogy a rögzített értéket befolyásolják a környező fényviszonyok és az érzékelőnek a színes felülethez való közelsége [16].



23. ábra: Színérzékelő [16]

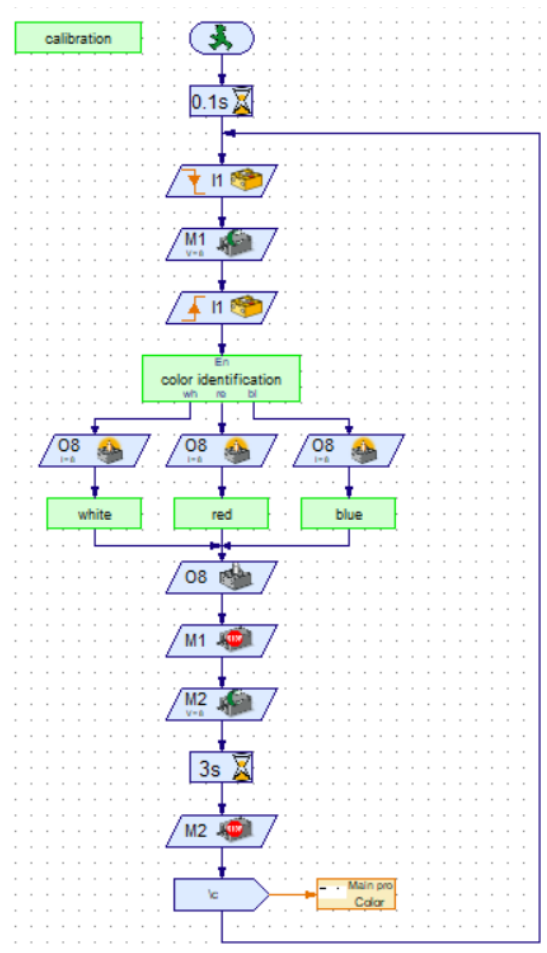
6.2.2 Gyári program és működés

A színválogató készre szerelt állomáshoz tartozik egy TXT Controller is, mely a működéshez szükséges vezérlő programot futtattja. Sajátos programozói környezet is tartozik a vezérlőhöz, ez a ROBO Pro.



24. ábra: TXT Controller [16]

Programnyelve hasonló az SFC-hez ami a programozható logikai vezérlők (PLC-k) grafikus programozási nyelve, mely az IEC 61131-3 szabvány részét képezi. Az SFC egy rendszerben vagy folyamatban a műveletek áramlását úgy ábrázolja, hogy azokat lépések és átmenetek sorozatába szervezi. A lépések olyan műveleteket tartalmaznak, amelyek akkor kerülnek végrehajtásra, amikor a lépés aktív, az átmenetek pedig a teljesítendő feltételek alapján szabályozzák az egyik lépésből a következőbe történő átmenetet.



25. ábra: ROBO Pro-ban megírt gyári program [16]

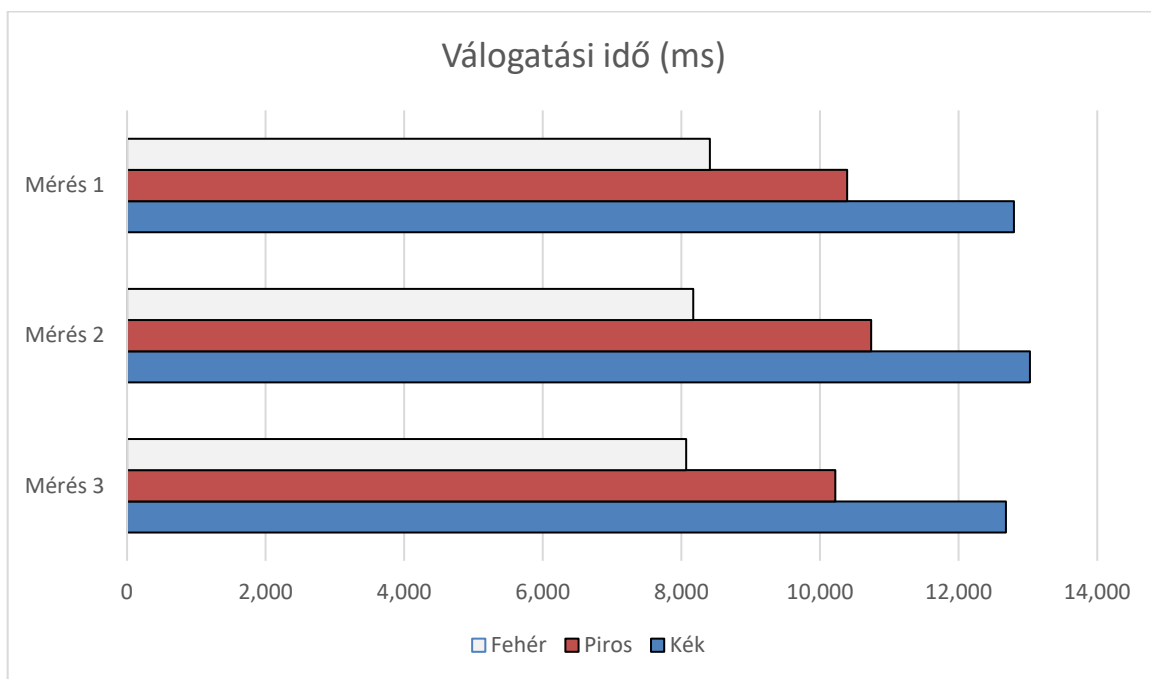
Ez a vizuális megközelítés a programozási folyamatot könnyebben hozzáférhetővé teszi, különösen a kezdők és a fiatalabb felhasználók számára, mivel lehetővé teszi számukra, hogy kézzelfoghatóbb módon lássák a programjuk menetét. A felület drag-and-drop jellege lehetővé teszi a felhasználók számára, hogy kirakós játékként rakják össze a programjukat, így a logika és a műveletek sorrendje világos és könnyen érthető lesz. A programfolyamat ábrázolásának ez a módszere különösen fontos a robotikában és az automatizálásban, ahol a műveletek sorrendje és időzítése gyakran kritikus fontosságú. A vizuális ábrázolás lehetővé teszi a felhasználók számára, hogy gyorsan felfogják az események sorrendjét, és megértsék,

hogy a különböző feltételek vagy események hogyan befolyásolhatják a program menetét [16].

A válogató állomás működése lényegében leolvasható a programjáról. Maga a folyamat egy 100ms-s késéssel indul, ezután az I1-re bekötött fototranzisztoros érzékelő falling edge pulzusára elindul a futószalag a megadott sebességgel. Majd amint elhagyta a munkadarab az első fénykaput tovább halad a fényérzékelő felé. Három színű munkadarab különválogatását tudja ellátni a megírt program ezek a fehér, piros és kék. Majd miután megállapította a vezérlő, hogy milyen színű munkadarab érkezett meg, akkor a megfelelő helyre eltárolja. A gyári program csak egy időben egy munkadarab szétválogatására képes.

6.2.3 Válogatási idő

A Fischertechnik színválogató állomás elemzése során megmértem az egyes munkadarabok színük - fehér, piros és kék - alapján történő válogatásához szükséges időt. Ez a mérés lehetővé tette, hogy megértsem a válogatási folyamat hatékonyságát és sebességét az egyes színek esetében.



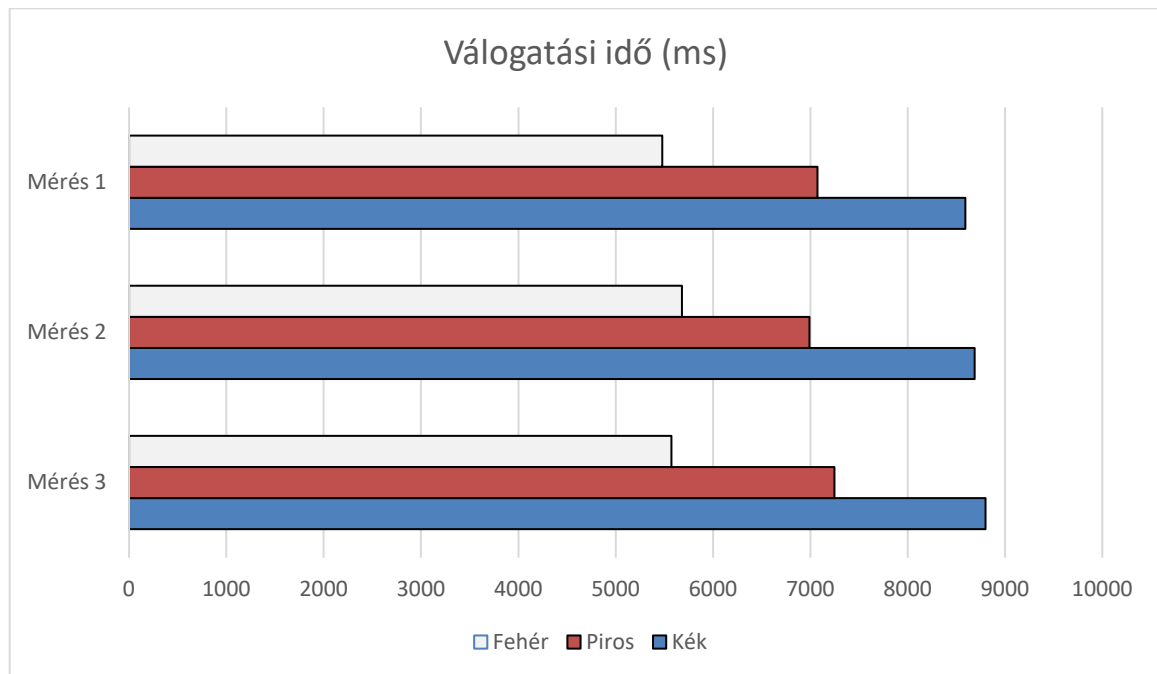
26. ábra: Az egyes munkadarabok válogatási ideje milliszekundumban mérve

A Fischertechnik színválogató állomáson belül a hatékonyság növelése és a válogatási idő csökkentése érdekében különböző lehetséges módosításokat lehet végrehajtani. Először is, a szállítószalag sebességének növelése azt eredményezi, hogy csökken az egyes munkadaraboknak a szortírozásához szükséges idő. Másodszor, a válogató állomás

működését irányító programozási kódot lehet finomítani azzal a céllal, hogy gyorsabbá váljon a rendszer mozgása, vagy növelni lehessen a hatékonyságot. A javasolt módosítások megvalósításával a végső cél a Fischertechnik színválogató mechanizmus általános hatékonyságának növelése, ezáltal az egyes munkadarabok válogatásához szükséges idő csökkentése, és ezzel párhuzamosan az állomás teljesítményének növelése.

6.2.3.1 Futószalag sebességének hatása a válogatási időre

A válogatási folyamat optimalizálása érdekében a szállítószalag sebességének növelésével végrehajtottam egy módosítást. Ennek a módosításnak az volt a célja, hogy csökkentsem az egyes munkadaraboknak szétválogatásának az idejét, végső soron az állomás általános hatékonyságának növelése érdekében. A módosítást követően újabb mérésorozatot végeztem, hogy felmérjem a megnövelt szállítószalag sebességének hatását az egyes színek - fehér, piros és kék - válogatási idejére. A mérés, módosítás és újramérés szigorú megközelítése kulcsfontosságú az elvégzett változtatások hatékonyságának mennyiségi értékeléséhez, és biztosítja, hogy a fejlesztések összhangban legyenek azzal a céllal, hogy növeljék az állomás áteresztőképességét, ugyanakkor megőrizzék a munkadarabok válogatásának pontosságát és megbízhatóságát.



27. ábra: Válogatási idő megnövelt futószalag sebesség mellett milliszekundumban

A mért adatok megvizsgálása után kiszámítható, hogy átlagosan 3326 ms javulást eredményez munkadarabonként a futószalag sebességének növelése.

6.2.3.2 *Módosítások a programban*

Az egyik kulcsfontosságú tényező egy olyan változtatás a programban, amely segítségével a válogató állomás képes több munkadarab egyidejű kezelésére a szállítószalagon. Ez a fejlesztés kétségtelenül optimalizálja a válogatási folyamatot, ezáltal jelentősen csökkentve a teljes válogatási időt. Mindenképpen meg kell azonban vizsgálni az ilyen kifinomult szoftvermódosítások megvalósíthatóságát a különböző programozási környezetekben és vezérlőkön. Elmélyedve a szükséges szoftvermódosítások bonyolultságában, nyilvánvalóvá vált számomra, hogy a programozható logikai vezérlő (PLC) a TXT vezérlőhöz képest megvalósíthatóbb platformként jelenik meg e fejlesztéshez. Ennek elsődleges oka a PLC-k képessége az összetett programozási struktúrák hatékony kezelésére, ami több munkadarab egyidejű kezeléséhez elengedhetetlen. A PLC-k robusztus architektúrája, valamint az ipari szabványokkal való kompatibilitásuk miatt jól alkalmazhatók a bonyolult programok végrehajtására, melyek kulcsfontosságúak a válogatási folyamat optimalizálásában. A szoftveres módosítás eredményét a későbbiekben mutatom be, a már digitalizált válogató folyamaton.

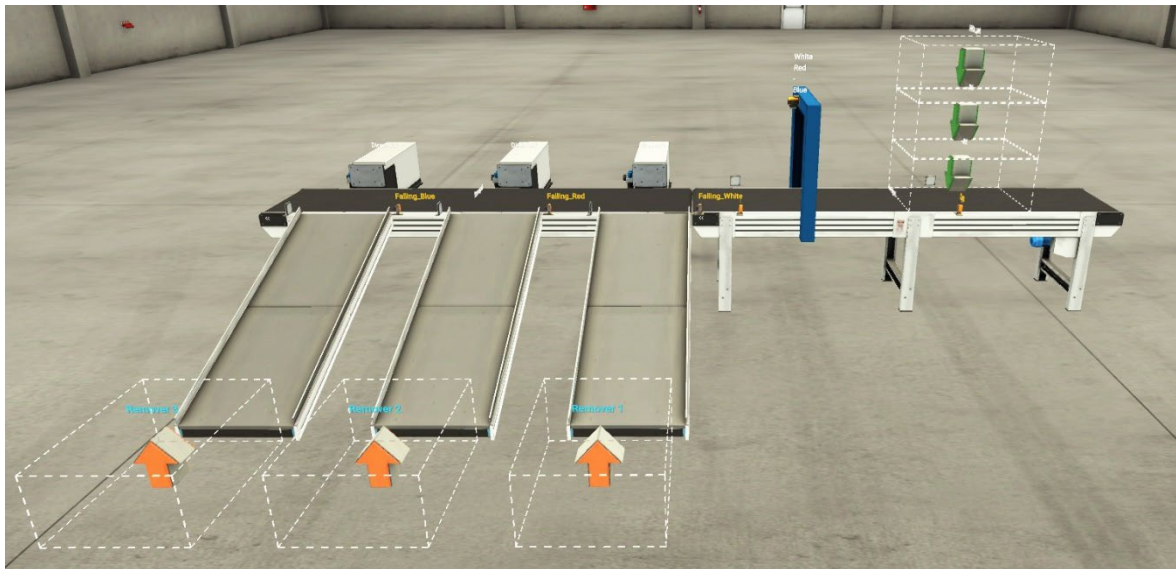
6.3 A válogató állomás megépítése és vezérlése digital lean szoftverben

Ebben a fejezetben bemutatom a válogatóállomás kivitelezését egy digitális lean szoftverkörnyezeten belül. Ez a vizsgálat kulcsfontosságú, mivel lehetővé teszi számomra, hogy a digitális lean eszközök innovatív képességeit kihasználva holisztikus szempontból optimalizáljam a válogatási folyamatot. A válogatóállomás digitális környezetben történő szimulálásával azonosítani tudom a lehetséges szűk keresztmetszeteket és a javításra szoruló területeket, megnyitva az utat egy hatékonyabb működés előtt. **Ez a megközelítés összhangban van a lean gyártás elveivel, amelyek a pazarlás kiküszöbölését és az értékteremtő tevékenységek fokozását hangsúlyozzák.**

A különböző eszközök közül úgy döntöttem, hogy először a Factory I/O-ban tervezem meg a szimulációt. Bár a Factory I/O 30 napos ingyenes próbaverziót kínál, ennek lejártá után költséges befektetéssé válik. Tekintettel a pénzügyi korlátokra és a személyre szabottabb megoldás iránti vágyra, úgy döntöttem, hogy saját szimulációs eszközt készítek a Godot játékmotor segítségével, amely nemcsak a kívánt rugalmasságot biztosította, hanem a Factory I/O-hoz kapcsolódó ismétlődő költségeket is megszüntette. Ez a váltás lehetővé tette számomra, hogy a szimulációt még inkább az én egyedi igényeimre szabjam, biztosítva ezzel a személyre szabottabb és hatékonyabb tervezési folyamatot. A Godot kiválasztásának

indoklása a sokoldalúságából és hozzáférhetőségéből eredt, amely lehetővé teszi a válogatási folyamat átfogó és részletes szimulációját. A projekt keretein belül többféle feladatot is elvégeztem: kódoltam, modelleztem, elemeztem, és még sok minden mást is, hogy a szimuláció megvalósulhasson. Ez a sokoldalú munka nem csupán a válogatási folyamat bonyolultságának mélyebb megértését segítette elő, hanem értékes gyakorlati tapasztalatot is nyújtott egy sokrétű projekt irányításában és végrehajtásában. A Godot játékmotorban a koncepciótól a megvalósításig tartó utat aprólékosan végig jártam biztosítva, hogy a válogatóállomás minden aspektusát pontosan ábrázoljam és elemezzem az optimális eredmények érdekében.

6.3.1 Factory I/O-s megvalósítás



28. ábra: Színválogató állomás digitális mása Factory I/O-ban (képernyő fotó)

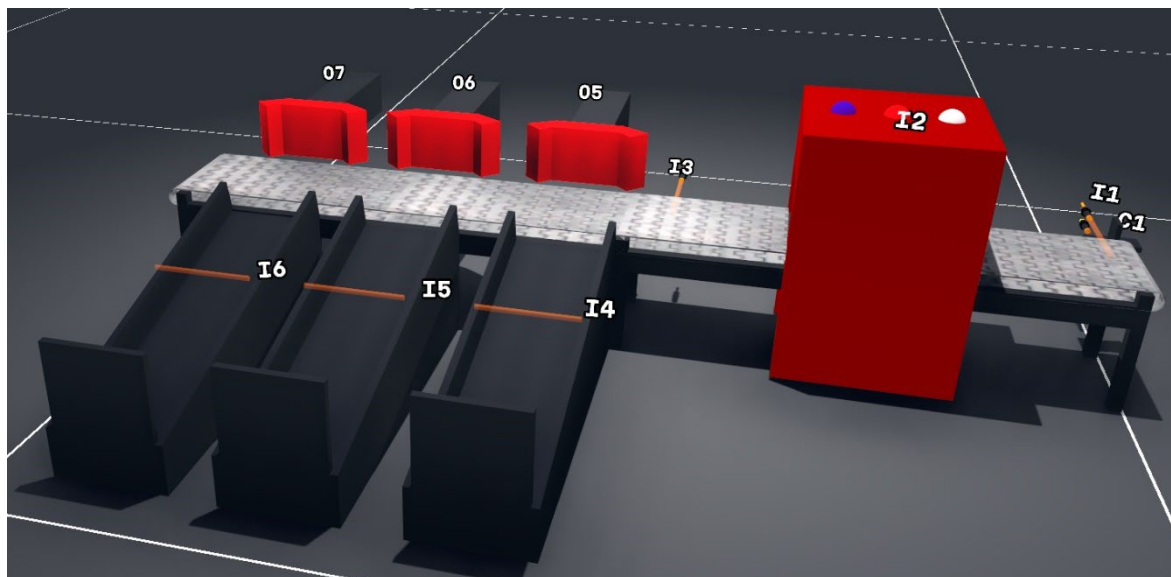
A válogatóállomás részletes ábrázolására törekedve a Factory I/O nevű programhoz fordultam, mivel jól ismert a valósághű ipari szimulációk készítésében. A fenti ábrán látható a Factory I/O környezetben tervezett gondos elrendezés. A válogatóállomás fizikai aspektusait emulálva, a szállítószalagoktól az érzékelőkig minden elemet beépítettem, hogy a valódi berendezést hűen leképpezem. A Factory I/O egy olyan robusztus platformot biztosított, ahol zökkenőmentesen felépíthettem és megjeleníthettem a válogatóállomás működését. A dinamikus környezet lehetővé tette különböző forgatókönyvek szimulációját, tesztelve a rendszer reakcióit különböző körülmények között. Az intuitív felhasználói felület, valamint az átfogó eszközkészlet megkönnyítette a hatékony tervezési folyamatot.

Bár a Factory I/O értékes előzetes vizualizációt nyújtott, korlátai hamarosan nyilvánvalóvá váltak számomra. A pénzbeli megfontolásokon túl - mivel a szoftver 30 napos ingyenes próbaverziója a végéhez közeledett, és a licencköltségek közeledtek - **a szoftver merevsége is kihívást jelentett** számomra. A platformból hiányzott az általam kívánt rugalmasság az egyedi testreszabásokhoz, ami korlátozta a személyre szabás azon mértékét, amelyet el szerettem volna érni. A korlátok melyek felmerültek a következők:

- **Korlátozott testreszabási lehetőségek:** A Factory I/O nem nyújtotta a projektemhez szükséges szintű személyre szabhatóságot. Nem rendelkezett azzal a képességgel, hogy egyedi szenzor típusokat és specifikus szállítószalag-konfigurációkat lehessen beépíteni, amelyek döntő fontosságúak voltak a válogatási folyamatmodellem pontos ábrázolásához.
- **Szoftverfelület és felhasználói élmény:** A felhasználói felület nem volt megfelelő a projektem sajátos munkafolyamatához. Hiányoztak olyan kulcsfontosságú funkciók, amelyek egyszerűsítették volna a tervezési folyamatomat, így a szoftver kevésbé intuitív és időigényes volt a sajátos igényeimnek megfelelően.
- **Anyagi tulajdonságok szimulációjának hiánya:** A szoftver nem szimulálta pontosan a speciális anyagtulajdonságokat, például a sűrűlódást, a súlyt és a tárgyak textúráját. Ez különösen korlátozó volt egy olyan projekt esetében, mint a színválogató rendszer, ahol a válogatott tárgyak fizikai tulajdonságai jelentősen befolyásolhatják a rendszer teljesítményét.
- **Nem támogatott az egyéni programozás, szkriptelés:** A Factory I/O nem támogatta az egyéni szkriptek vagy programozási nyelvek használatát, ami korlátozta a szoftver szabványos képességeit meghaladó összetett logika vagy algoritmusok megvalósításának lehetőségét.
- **Korlátozott "Community Support":** A Factory I/O egyik fő korlátja a felhasználói közösség támogatásának hiánya volt. Az összetett vagy egyedi projektelemek esetében egy élénk felhasználói közösség felbecsülhetetlen értékű a hibaelhárítás és a tudásmegosztás szempontjából. Sajnos a Factory I/O felhasználói közössége nem volt sem elég kiterjedt, sem elég aktív ahhoz, hogy a szükséges segítséget és betekintést nyújtsa, ami kihívást jelentett az egyes problémák megoldása vagy a mások tapasztalataiból való tanulás terén.

Ezen korlátozások a pénzügyi szempontokkal együtt arra készítetett, hogy alternatív megoldásokat keressek. Következésképpen elindultam egy olyan egyéni szimuláció megtervezése felé, amelyet kifejezetten a projekt egyedi követelményeihez igazítottam.

6.3.2 Saját szimulációs megvalósítás



29. ábra: Színválogató állomás digitális mása Godot-ban (saját)

Az ábra a Godot játékmotoron belül a válogatóállomás újra alkotására tett aprólékos munkámat mutatja be. Mint látható, minden elemet - a szállítószalagoktól a válogatótárolókig - úgy készítettem el, hogy mind funkcionalitásban, mind esztétikailag szorosan utánozzam a valós válogató állomást. Míg a szimuláció a Godotban készült, a tényleges folyamatvezérlést egy már korábban említett Codesys PLC irányította. Ez a hibrid megközelítés ötvözte a Godot grafikus képességeit, a PLC robusztus és ipari szintű vezérlésével. Ennek a konfigurációnak az egyik fő előnye az volt, hogy változatos válogatási scenáriókat tudtam tesztelni. A Godot szkriptelési lehetőségeit a Codesys PLC vezérlési logikája mellett kihasználva különböző körülményeket tudtam szimulálni és szorosan nyomon követni a rendszer reakcióját. Ez kulcsfontosságú volt a javításra vagy módosításra szoruló területek proaktív feltárásában. A digitális más megvalósítása közben törekedtem arra, hogy a folyamat **ugyanúgy működjön**, mint a valós mása. Így az optimalizálásra tett kísérletek ugyanazt az eredményt adják a valóságban, mint a szimulációban.

6.3.3 Vezérlő logika újragondolása

Az automatizált rendszerek területén a több munkadarab egyidejű feldolgozása a szekvenciális kezeléssel szemben jelentősen növelheti az átfutási kapacitást és a

hatékonyságot. Felismerve ezt a potenciált, elkezdtem a Godot által tervezett szimulációs környezetben lévő vezérlő logika újragondolását és újra tervezését. A kihívás az volt, hogy olyan logikát dolgozzak ki, amely lehetővé teszi több munkadarab egyidejű mozgását a szállítószalagon anélkül, hogy veszélyeztetné a válogatási folyamat teljességét. Ez egy komoly koncepcionális újra tervezési folyamatot jelentett, amelynek során a meglévő szekvenciális logikát elemeire bontottam és egy összetettebb, de hatékonyabb párhuzamos feldolgozási modellé raktam össze.

Ennek az újra tervezésnek a központi eleme egy olyan tömbstruktúra, a "BoxArray" bevezetése volt, amely képes egyszerre több doboz követésére, miközben azok a szállítószalagrendszeren haladnak át. Ez a tömb, valamint egy sor Boolean flag és integer számláló biztosítja a rendszer állapotának valós idejű reprezentációját.

```
1  TYPE Box:
2  STRUCT
3      Color: INT := 0; // 0: No color, 1: White, 2: Red, 3: Blue
4      Position: INT := 0; // Number of gear rotations since detection
5      StartCountingIndividual: BOOL := FALSE;
6  END_STRUCT
7  END_TYPE;
8
```

30. ábra: Box tömbstruktúra program részlet (saját)

A kialakított rendszer dinamikusan igazítja a szállítószalag működését a munkadarabok valós idejű áramlásához alkalmazkodva, optimalizálva a szortírozás pontosságát és időzítését. A szortírozott dobozok elszállítása zökkenőmentesen történik, a rendszer a szállítószalag terhelésének változásaihoz igazodva kalibrálódik újra. Ez lehetővé teszi a munkadarabok folyamatos áramlását szükségtelen megszakítások vagy állásidő nélkül. Összességében az átstrukturált logika növeli a válogatóállomás kapacitását és alkalmazkodóképességét, lehetővé téve egy gyorsabb és hatékonyabb válogatási folyamatot, amely precízen képes kezelni a változó munkadarab-forgatókönyveket.

Ahhoz, hogy a végrehajtott fejlesztést összetudjam hasonlítani az eredeti programmal, egy válogatási forgatókönyvet kellett felállítanom. Megvizsgáltam, hogy mennyi időbe telik az állomásnak elválogatni 3-3 db munkadarabot az egyes színekből. Összesen 9 munkadarab válogatását vizsgáltam. Az eredeti beállítások és a futószalag sebesség növelése mellett a kék munkadarabok elválogatása átlagosan 8694 ms-ot, a piros munkadarabok 7103 ms-ot míg a fehér 5577 ms-ot vett igénybe. Ezen beállítások és vezérlési logika mellett a 9 munkadarab

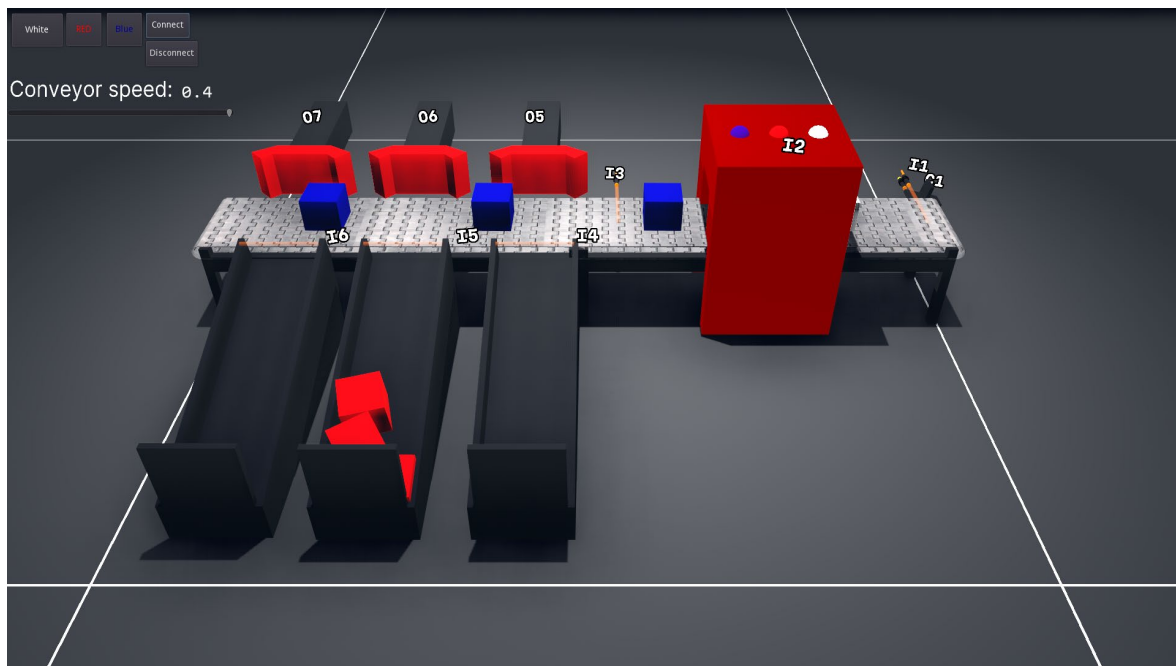
elválogatása legoptimálisabb esetben is csak $8694 * 3 + 7103 * 3 + 5577 * 3 = 64\,122$ ms, azaz 64 másodperc.

Az optimalizált programmodosítások és a futószalag sebességének növelésével az alábbi eredményeket értem el.



31. ábra: Válogató állomás programja [mérés 1, mérés 2, mérés 3] (saját)

Átlagosan 25520 ms azaz 25,52 másodperc alatt történt meg a 9 munkadarab elválogatása.



32. ábra: Válogató állomás egyszerre több munkadarabbal (saját)

A válogatóállomás programjának optimalizálása és a szállítószalag sebességének növelése jelentős teljesítményjavulást eredményezett. Eredetileg 9 munkadarab (színenként 3 darab) szétválogatása összesen 64,122 másodpercet vett igénybe. A fejlesztésekkel ugyanezt a feladatot mindössze 25,52 másodperc alatt sikerült elvégezni.

Ez 60,2%-os hatékonyságnövekedést jelent, ami jelentősen csökkentette a válogatási folyamathoz szükséges időt. Az, hogy ugyanazt a munkadarabszámot az eredeti idő nagyjából 40%-a alatt lehet szétválogatni, az állomás működési teljesítményének jelentős javulását mutatja, és a végrehajtott programmodosítások és a szállítószalag sebességének beállítása hatékonyságát bizonyítja.

Összegzés

Ez a projekt a Lean gyártási elvek és a digitális szimulációs technológiák integrációját vizsgálja az ipari folyamatok hatékonyságának növelése érdekében. A dolgozat a különböző ipari folyamatok áttekintésével kezdődik, és a Lean gyártás alapelveivel foglalkozik, hangsúlyozva a hulladékcsökkentést és a hatékonyság növelését. Ezután bemutatja a Digital Lean koncepcióját, kiemelve a digitális hulladék megjelenését és a kiber-fizikai termelési rendszerekre (CPPS) mint a termelés jövőjére való áttérést. A dolgozat jelentős részét az ipari folyamatok szimulációjának szenteltem, felvázolva a szimulációk különböző típusait, különös tekintettel a diszkrét eseményű és a folytonos szimulációkra.

Kutatásom központi eleme egy egyedülálló szimulációs program kifejlesztése a Godotban. Ez az aprólékos munka egy sokoldalú, különböző folyamatszimulációkra alkalmas keretrendszer eredményezett, amely túllépve a projekt hatókörén, egy strukturált, más ipari helyzetekre adaptálható és skálázható sablont kínál. Ez a keretrendszer értékes eszközzé válhat a jövőbeni, egyedi szimulációkat igénylő projektek számára, ami jól szemlélteti az innováció szerencsés természetét a konkrét kihívások megoldásában, miközben szélesebb körű alkalmazások és megoldások előtt nyit utat.

A projekt tetőpontja a válogatóállomás vezérlő logikájának optimalizálása volt. Az eredeti válogatási folyamatot alaposan összehasonlítottam az újonnan kifejlesztett eljárással, 9 munkadarab válogatásának ütemezésével. Az optimalizált program és a megnövelt szállítószalagsebesség 60,2%-os hatékonyságjavulást eredményezett, a válogatási időt 64,122 másodpercről 25,52 másodpercre csökkentve, ezzel is kiemelve a Lean elvek és a digitális eszközök kombinálásában rejlő jelentős lehetőségeket az ipari folyamatok racionalizálása érdekében.

Irodalomjegyzék

- [1] G. Kovács és T. Péter, Termelési folyamatok bemutatása és vizsgálata, Miskolc: Miskolci Egyetem, 2015.
- [2] G. Péczely, A Lean alternatív értelmezésének értékelése a hazai feldolgozóipari vállalatok körében, Veszprém: Pannon Egyetem, 2017.
- [3] M. F. William, Lean Manufacturing, New York: The St. Lucie Press, 2001.
- [4] K. Demeter és I. Jenei, A Lean menedzsment és a versenyképesség kapcsolata, Budapest: Versenyképesség Kutató Központ, 2011.
- [5] D. Romero, "Digital Lean Cyber-Physical Production," Springer Nature Switzerland AG, London, 2018.
- [6] T. Uhlemann, C. Lehmann and R. Steinhilper, "The Digital Twin: Realizing the Cyber-Physical Production System for Industry 4.0," *Procedia CIRP*, vol. 61, pp. 335-340, 2017.
- [7] M. L. Averill, Simulation Modeling and Analysis (5. kiadás), Tucson: Averill M. Law & Associates, Inc., 2013.
- [8] G. Szűcs, Diszkrét szimuláció matematikai alapjai, Budapest: BCE Operációkutatás Tsz., 2007.
- [9] D. Nagy, Paraméterezhető DES modellek fejlesztése Plant Simulation szoftverrel, Miskolc: Miskolci Egyetem, 2015.
- [10] J. Butcher, Numerical Methods for Ordinary Differential Equations, New Zealand: The University of Auckland, 2008.
- [11] J. D. Anderson, Computational Fluid Dynamics: The Basics with Applications, New York: McGraw-Hill, 1995.

- [12] J. Banks, Handbook of simulation, Kanada: John Wiley & Sons, 1998.
- [13] D. Nemes, T. Pálfi and S. Hajdú, “Vehicle Dynamic Simulation Possibilities Using AVL,” *International Journal of Engineering and Management Sciences*, vol. 5, no. 2, pp. 309-323, 2020.
- [14] S. Spieckermann, K. Gutenschwager, H. Heinzl and H. Vob, “Simulation-based optimization in the automotive industry - A case study on body shop design,” *Transactions of The Society for Modeling and Simulation International - SIMULATION*, vol. 75, 2000.
- [15] R. Conway, W. Maxwell, J. McClain and L. Thomas, “The Role of Work-in-Process Inventory in Serial Production Lines,” *Operations Research*, vol. 36, no. 2, pp. 229-241, 1988.
- [16] „Fischertechnik,” Fischertechnik, [Online]. Available: <https://www.fischertechnik.de/en/industry-and-universities/technical-documents/old-assortment/sortierstrecke-mit-farberkennung-9v>. [Hozzáférés dátuma: 10. 09. 2023.].
- [17] „Factory I/O,” [Online]. Available: <https://factoryio.com/features>. [Hozzáférés dátuma: 17. 09. 2023.].
- [18] „Shapr3D,” [Online]. Available: <https://www.shapr3d.com/company/about-shapr3d>. [Hozzáférés dátuma: 07. 09. 2023.].
- [19] “codesys control win,” CODESYS, [Online]. Available: <https://store.codesys.com/en/codesys-control-win-sl-bundle.html>. [Accessed 07. 09. 2023.].
- [20] T. Török és E. Barta, Fém és szerves bevonattechnológiák, Miskolc: Miskolci Egyetem, 2013.
- [21] J. Casad, TCP/IP in 24 Hours, Sams Teach Yourself, Carmel: Sams Publishing, 2017.

- [22] H. Vargas, R. Heradio and M. Donoso, “Teaching automation with Factory I/O under a competency-based curriculum,” *Multimedia Tools and Applications*, vol. 82, p. 19221–19246, 2022.
- [23] “Godot Engine,” Godot, [Online]. Available: <https://godotengine.org/>. [Accessed 07. 09. 2023].