Multidiszciplináris tudományok, 10. kötet. (2020) 2 sz. pp. 413-420 **https://doi.org/10.35925/j.multi.2020.2.46**

AUTONÓM JÁRMŰMODELL FEJLESZTÉSE

Anga Dávid
hallgató, Miskolci Egyetem, Automatizálás és Infokommnikációs Intézet
3515 Miskolc, Miskolc-Egyetemváros, e-mail: anga.david@student.uni-miskolc.hu

Koba Máté
egyetemi tanársegéd, Miskolci Egyetem, Automatizálás és Infokommnikációs Intézet
3515 Miskolc, Miskolc-Egyetemváros, e-mail: autkmate@uni-miskolc.hu

Absztrakt

Az autóiparban egyre fontosabb és elvártabb az önvezető vagy legalább vezetéstámogató funkciók beépítése. Ezért is vezették be a Formula Student versenysorozatban is az autonóm versenyszámokat, melyek során az autónak saját „meglátásai" alapján kell döntéseket hoznia az irányításban. Egyelőre a Formula Racing Miskolc csapata nem rendelkezik erre kész megoldással. Ennek a fejlesztésében az első lépés volt egy járműmodell fejlesztése mellyel lehetőség nyílik az első tesztek elvégzésére. A következőkben az ehhez kapcsolódó irodalomkutatást és kezdetleges eredményeket, észrevételeket összegzem.

Kulcsszavak: *önvezető jármű, gépi látás, képfeldolgozás, Formula Student*

Abstract

In the automotive industry, the incorporation of self-driving or at least driving support functions is becoming increasingly important and expected. This is why autonomous races have been introduced in the Formula Student race series, during which the car has to make decisions in control based on its own "insights". For the time being, the Formula Racing Miskolc team does not have a ready-made solution. The first step in developing this was to develop a vehicle model that would allow the first tests to be performed. In the following, I summarize the related literature search and initial results and remarks.

Keywords: *self-driving vehicle, machine vision, image processing*

1. Bevezetés

Az utóbbi években az autóipar óriási mértékű fejlődésen ment át vezetéstámogató rendszerek tekintetében. Gondolok itt például a távolságtartó tempomatra vagy a sávtartó elektronikára, holttérfigyelő rendszerre. Ezen rendszerek összehangolt működése elengedhetetlen ahhoz, hogy egy jármű képes legyen magát irányítani és az úton maradni. Az ehhez szükséges rendszerekből többféle megoldás is létezik, van amelyik kamerák segítségével „látja" a környezetét és különböző képfeldolgozó algoritmusok dolgoznak a háttérben, valamint olyan is amelyik sokféle szenzor segítségével érzékeli a világot maga körül. E kettő közül én az utóbbit választottam a saját megoldásomhoz, melyet e cikkben részletezem.

2. Formula Student

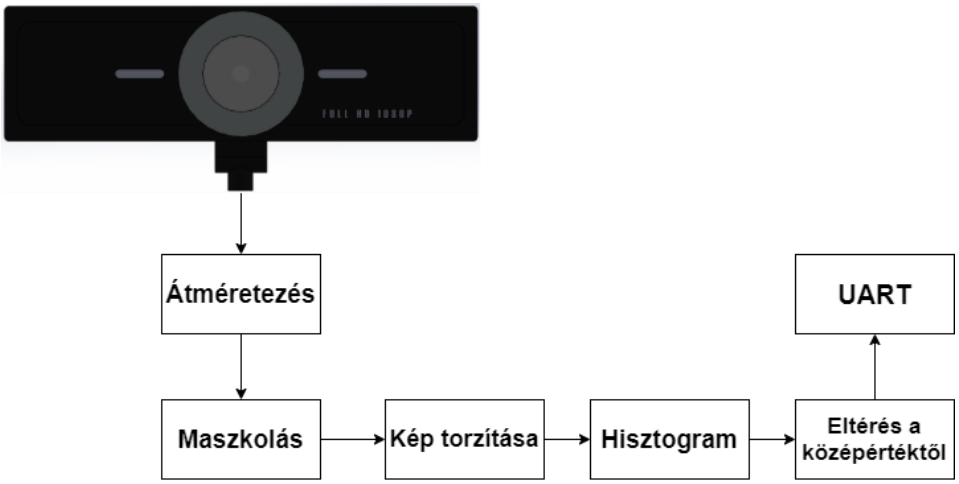
A Formula Student egy olyan versenysorozat, amelyben aktív egyetemi hallgatók mérhetik össze tudásukat „éles" helyzetben. A feladat az, hogy egy versenyautót építsenek a csapatok a szabályzatnak megfelelően. A verseny során több úgynevezett statikus és dinamikus versenyszám is van, melyek során

például pontozzák a mérnöki megoldásokat vagy épp az üzleti tervet. A versenyszámok során összegyűjtött pontok összegzéséből kerül felállításra a csapatok sorrendje. A versenysorozat életében újdonságnak számít az önvezető (Driverless) kategória, mely során nagyban hasonló versenyszámokat kell teljesítenie az autónak. [1] Ezeknek a teljesítéséhez rengeteg képfeldolgozással és egyéb gépi látással kapcsolatos ismeretre, valamint tesztekre szükség van, hogy segítségükkel egy stabilan működő megoldás születhessen. A fejlesztés kezdeti szakaszához készítettem egy modellautót a teszteléshez.

Ebben az esetben a legfontosabb kritérium a költséghatékonyság volt, mivel annak célja csak annyi volt, hogy ismereteimet bővítsem a képfeldolgozás területén. Ezzel gyorsítható volt a folyamat és a tesztelések is könnyebben elvégezhetőek, mint ha egy valós versenyautón kellene ezeket kivitelezni. A végleges megoldás ettől sokkal összetettebb algoritmusokat fog alkalmazni, melyhez nagyobb számítási kapacitású hardverre lesz szükség. Jelen esetben egy egyszerűsített képfeldolgozás miatt elegendő volt egy Raspberry Pi 4 Model B típusú egység is ennek elvégzésére.

3. Képfeldolgozás

A képfeldolgozás egy egyszerűsített megoldást kapott, melynek segítségével egy fehér sáv megtalálható és azt képes követni a modellautó. A versenyautóra szánt megoldás ezen a hardveren még nem fejleszthető, ugyanis ott szükséges alakzat- és színfelismerés is, mivel a pálya bójákkal kerül kijelölésre és van olyan versenyszám mely esetében figyelni kell azok színét is. A modellautón történő tesztelés során szerzett tapasztalatokkal nagyobb rálátás nyerhető a szükséges számítási teljesítményre és annak legmegfelelőbb kiaknázásra. Továbbá a képfeldolgozási algoritmusok megismerésével gyorsítható a megfelelő kiválasztása is.



1. ábra. *Képfeldolgozás blokkvázlata*

A fejlesztés Python programnyelven történik és a képfeldolgozáshoz a nyílt forráskódú programozási könyvtárat, az OpenCV-t használom. Az ebben található függvények segítségével történik a feldolgozás megvalósítása. [2]

3.1 Kamera

Képalkotó berendezésnek egy USB-s webkamerát választottam, amely 1920x1080 pixel felbontású képanyag létrehozására képes. A fókusztávolság manuális a lencse körül található gyűrű segítségével

állítható be. Az egység nagy látószöggel rendelkezik, amely előnyös ebben az esetben, mivel a modellautó teljes szélessége megfigyelhető segítségével.

3.2 Átméretezés

A kamera felbontása ebben az esetben túl nagy, így ezt a Raspberry Pi nem tudna megfelelő sebességgel feldolgozni. Ennek elkerülése céljából a beérkező képet átméretezéssel kezdem, melynek a felbontása ezután 480 x 240 pixel lesz. Ezzel jelentősen csökkenthető a feldolgozás sebessége és mivel ebben az esetben nem szükséges a nagy felbontás, így az eredményt ez nem befolyásolja negatívan.

3.3 Maszkolás

A beérkező kép szürkeárnyalatossá alakítása után az úgynevezett „Tresholding” módszerrel a képet bináris képpé alakítottam, ami azt jelenti, hogy csak fehér és fekete pixelekből épül fel, így egy monokróm képet kapok. A végeredmény függ az adott fényviszonyoktól, így mivel manuálisan lett az adottakhoz beállítva, ezért ha máshol történik a tesztelés akkor azt módosítani szükséges.

2. ábra. *A monokróm (balra) és az eredeti kép (jobbra)*

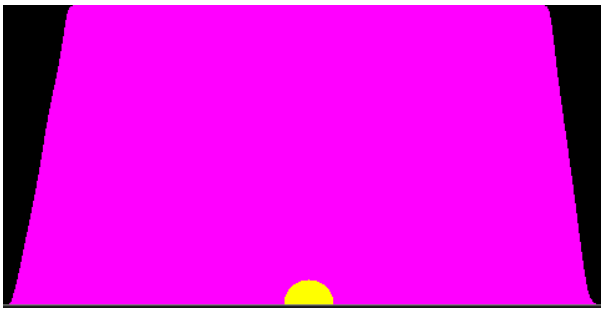
3.4 Kép torzítása

Az így kapott kép segítségével már végezhetőek számítások, de mivel a 2. ábrán is látható módon a sáv végig látható, így kanyar esetén nehezen lenne az érzékelhető. A gyorsabb érzékeléshez torzítva a képet „felülnézethez” hasonló módot hozok létre. Ezzel a sáv vonalvezetésében történő apróbb eltérések is észrevehetők. A 3. ábrán láthatóak szerint néz ez ki, amelyen a sáv mellett található keskeny fekete terület fontos szerepet tölt be a képfeldolgozás során.

3. ábra. *Torzított kép*

3.5 Hisztogram

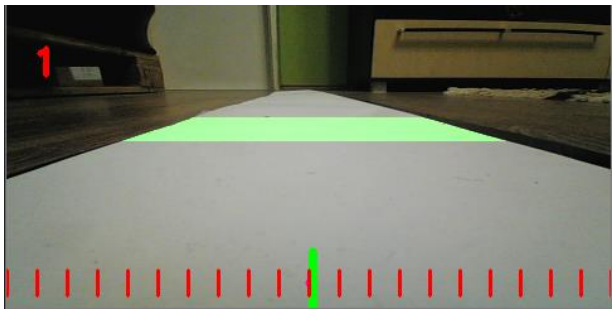
Az így kapott torzított képen hisztogram segítségével részekre bontva vizsgáljuk, hogy hol mekkora mértékű fehér szín (vagyis magas érték) található. A 4. ábrán az így kapott kép látható, melyen már a hisztogram értéknél kapott 0 érték is fel lett tüntetve egy sárga ponttal.



4. ábra. *Hisztogram eredménye*

3.6 Eltérés

A 4. ábrán bejelölt értéktől való eltérés kiszámítása után azt már lehet használni az irányításra. A végeredményt egy skálán szemléltettem, amely az eredeti képre lett ráhelyezve és a számérték is felírásra került arra (lásd 5. ábra). Az itt látható zöld vonal jelzi a kép közepét, valamint a zöld terület a feldolgozott területet.



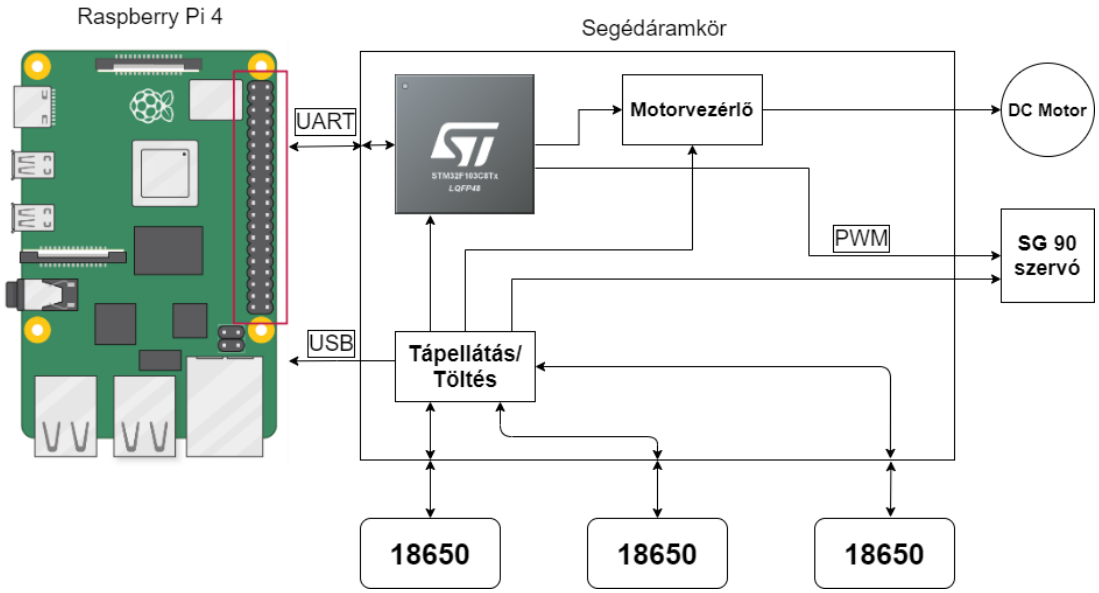
5. ábra. *Képfeldolgozás végeredménye*

Az így kapott érték segítségével már irányítható a modellautó, de ehhez szükséges egy segédáramkör mellyel feldolgozható az UART-on kapott érték és vezérelhetőek a motorok. Ezen felül biztosítja a táp-ellátást a Raspberry Pi és a motorok számára is.

4. Segédáramkör

Önmagában egy Raspberry Pi képes sokféle feladat ellátásra, de mivel egy alapvetően multimédiás célokra fejlesztett központi feldolgozóegységével operál, így szükséges volt egy olyan kiegészítő áramkörre, amely az alacsonyabb szintű vezérlési feladatoknak dedikálva kerül kialakításra.

A Raspberry Pi 4-ből UART-on keresztül kerül továbbításra a képfeldolgozás értéke, melyet egy mikrovezérlő segítségével feldolgozok, majd megfelelő PWM jellé kerül átalakításra a 7. ábra szerint.



6. ábra. *Segédáramkör blokkvázlata*

A 6. ábrán látható több egység is a segédáramkörön belül, melyek a következők szerint alakulnak:

- Motorvezérlő: Ez egy L293D nevű egység, mely segítségével az egyenáramú motor működtethető akár mindkettő irányban is. Ennek a vezérlését a mikrovezérlő látja el.
- SG90 szervó: Ennek működtetésének a feladatát is a mikrovezérlő látja el és ennek segítségével irányítható a modellautó. A 7. ábrán feltüntetettek szerint változik a kitöltési tényezője a PWM jelének.
- 18650 akkumulátorcellák: A tápellátás biztosítására 3 darab 18650-es elnevezésű akkumulátor cellát használok soros kapcsolásban.
- Tápellátás/Töltés: Ezt a blokkot több egység összevonásával hoztam létre. A töltésért cellánként felel egy töltésvezérlő áramkör. A tápellátást pedig három külön áramkör látja el, kettő darab 5 V-os és egy 3,3 V-os. Az 5 V-os áramkörök közül az egyik a Raspberry Pi ellátáshoz van dedikálva.

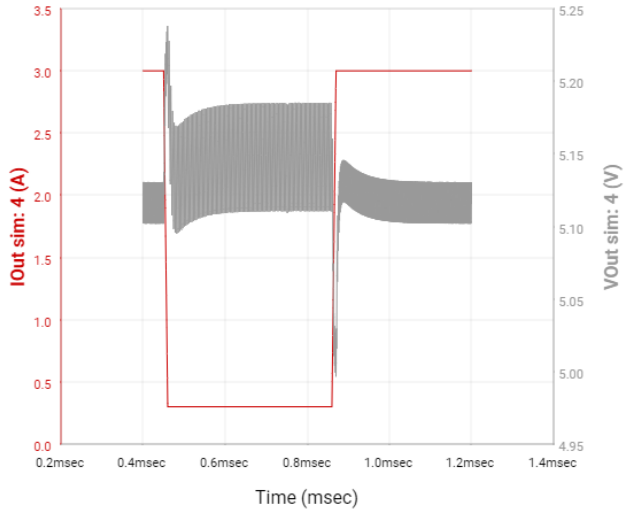
7. ábra. *PWM jel függvénye a képfeldolgozás értékétől*

4.1 Tápellátás

A tápáramkör tervezésekor nagy hangsúlyt fektettem arra, hogy a legmegfelelőbb legyen a Raspberry Pi számára, ezért több szimulációt is lefuttatásra került, melyekkel vizsgáltam ezt. Különböző kritériumok merültek fel, melyekre a kapott áramkör a legkevesebb külső alkatrész segítségével a legolcsóbban alkalmas volt.

A következő tesztek lettek elvégezve [3]:

- Felfutási idő: Ezzel a szimulációval azt vizsgáltam, hogy menyi idő szükséges az áramkör feszültség alá helyezése után a kimenetén az 5 V-os szint eléréséhez. A Raspberry Pi gyári tápegységének ezen értéke maximum 100 ms lehet, míg az általam kapott érték 0,94 ms lett. Ezzel tehát ennek a kritériumnak eleget tesz.
- Terhelésvizsgálat: Ezen teszt azért volt szükséges, mert hirtelen terhelésváltásnál fenn állt a veszélye annak, hogy leáll a Raspberry Pi a feszültségérték „beesése" miatt. Ám a szimuláció azt mutatja, hogy nem eshet be 5 V alá a legnagyobb áramfelvétel esetén sem.
- Bemeneti feszültségváltozás: Ennek a vizsgálatnak az elvégzésére az akkumulátorcellák merülése miatt volt szükség, mivel az 12,6 – 9 V közötti feszültségérték is lehet, így minden esetben stabil 5 V kell legyen a kimeneti feszültség, melyet a szimuláció szerint teljesíteni képes.

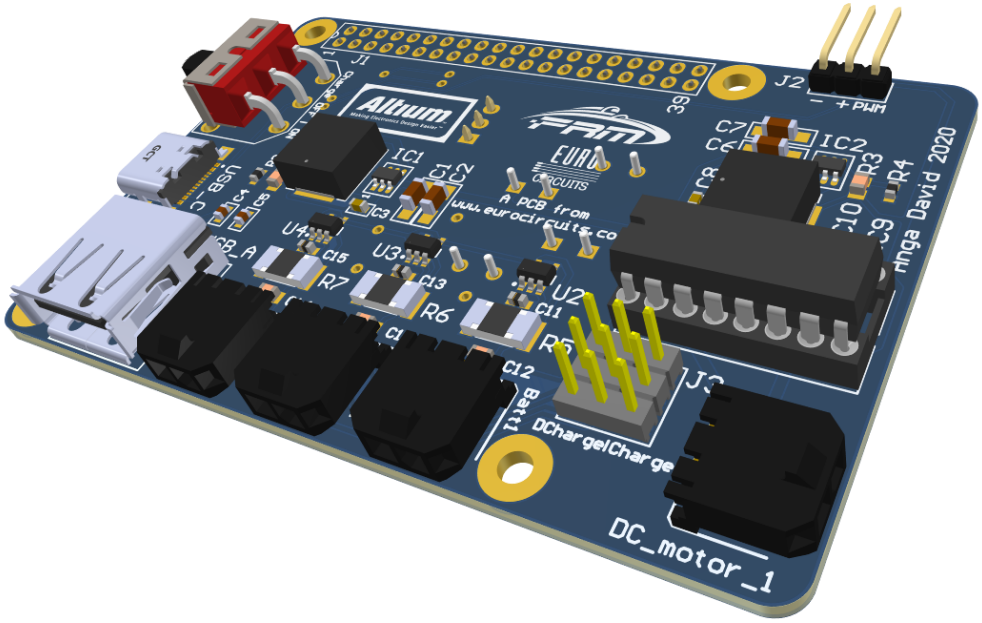


8. ábra. *Terhelésvizsgálat*

4.2 Nyomtatott áramköri terv

A terv készítésénél a Raspberry Pi alaterületével megegyező területben szabtam magamnak korlátot, mivel nem szerettem volna, hogy túllógások legyenek a végleges összeszerelésnél, így a 6.ábrán is látható végeredményt kaptam a huzalozás elvégzése után.[4] [5] [6]

Elhelyeztem még a szervómotor számára egy három érintkezőből álló tüskesort és egy ugyanilyet a mikrovezérlő programozójának, amely a NYÁK hátoldalán található. Az alkatrészek elhelyezésekor a lehető leghatékonyabb megoldást helyeztem előtérbe, az adott csatlakozóhoz tartozó áramkör közvetlenül az mögé helyeztem el, ám a kis méret miatt és az alkatrészek száma miatt így is bonyolult volt azok huzalozása.



9. ábra. *Segédáramkör 3D terve*

Az akkumulátorok töltésére szolgál a C típusú USB aljzat, valamint az A típusú USB aljzaton keresztül táplálható meg a Raspberry Pi, melynek tetejére kerül az egész segédáramkör. A kapcsoló segítségével indítható meg a cellák töltése és a 3 x 3-as tüskesor segítségével választható meg a cellák soros kapcsolása.

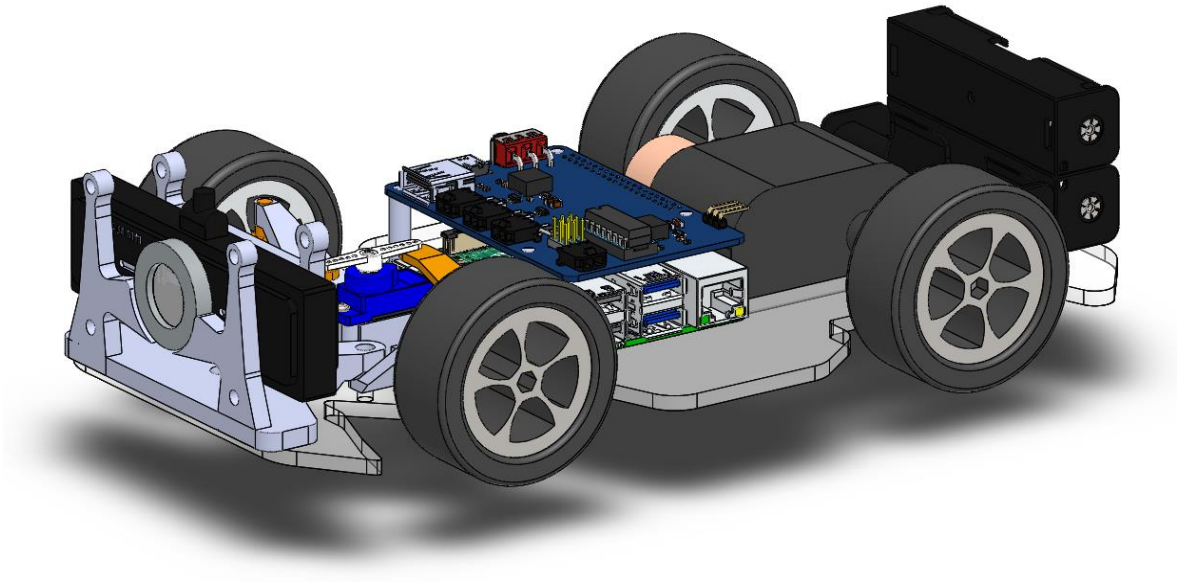
5. Irányítandó modellautó

A terv megvalósításához szükséges volt egy olyan autó, amelyet könnyen tudok majd irányítani és elfér rajta minden, amit szeretnék rátenni, így nem egy meglévő RC autó mellett döntöttem, hanem egy sajátot terveztem. Erre azért volt szükség, hogy a modellautó a lehető legjobban hasonlítson tulajdonságaiban a leendő önvezető Formula Student autóhoz, valamint a saját igényeimnek megfelelő legyen a kialakítás.

A tervezés elkezdésekor a meglévő alkatrészeim (Raspberry Pi, hátsó hajtás) méretei adtak kiindulópontot és meghatározóak voltak a teljes megvalósítás során. Alapul egy plexi lap szolgál, melyet lézer segítségével vágtak ki. A többi alkatrészt 3D nyomtató segítségével készült el. A modell összeszereléséhez szabványos méretű M3 és M4-es csavarokat választottam, melyek segítségével megfelelő stabilitás érhető el a felszerelt alkatrészek tekintetében, továbbá megoldott a szerelhetőség is.

Mivel a képfeldolgozás fontos részét képezi a kamera, ezért azt a lehető legstabilabban szerettem volna rögzíteni, ezért volt szükség a 10. ábrán megfigyelhető tartókonzolra. Az ezen látható furatokon keresztülhaladó menetesszárakkal kerül pozicionálásra a kamera.

Az így kapott modellautó segítségével egy rendezett tesztelési járművet kaptam, amely kevésbé érezteti magáról azt, hogy prototípus.



10. ábra. *Modellautó 3D terve*

6. Összefoglalás

A modell fejlesztésével elérhetővé válik a csapat számára egy olyan járműmodell mely segítségével könnyebben modellezhető az önvezetés és közelebb lehet kerülni a képfeldolgozás megismeréséhez. A végleges megoldás ettől nagyban eltérő lesz, de tapasztalatszerzés tekintetében nagyon előnyös egy ilyen modell. Ezen felül prezentálási szerepkörben is jó szolgálatot tud majd tenni.

7. Köszönetnyilvánítás

A cikkben ismertetett kutató munka az EFOP-3.6.1-16-2016-00011 jelű „Fiatalodó és Megújuló Egyetem – Innovatív Tudásváros – a Miskolci Egyetem intelligens szakosodást szolgáló intézményi fejlesztése” projekt részeként – a Széchenyi 2020 keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Irodalom

[1] Formula Student Rules 2020, 2020. [Online]. Available: https://www.formulastudent.de/fileadmin/user_upload/all/2020/rules/FS-Rules_2020_V1.0.pdf
[2] OpenCV, November 2012. [Online]. Available: https://en.wikipedia.org/wiki/OpenCV
[3] Webenc Power Designer, Texas Instruments, 2020. [Online]. Available: https://webench.ti.com/power-designer/switching-regulator/simulate/2
[4] IPC-2152, https://www.ipc.org/ContentPage.aspx?pageid=IPC-2152-Standard-for-Determining-Current-carrying-Capacity-in-Printed-Board-Design-Released
[5] Altium, online: https://resources.altium.com/p/using-ipc-2152-calculator-designing-standards
[6] Raspberry Pi, 28. 05. 2020. [Online]. Available: https://en.wikipedia.org/wiki/Raspberry_Pi