

FELADATKIÍRÁS

Jelenetértelmezés megvalósítása autonóm járművekhez mély neurális hálók segítségével A gépi tanulás új módszerei az intelligens érzékelés számos területét forradalmasították az elmúlt évtizedben. Ezen módszerek közül külön figyelmet érdemel a mély tanulás (Deep Learning), amely a gépi tanulás legtöbb területén state of the art megoldásnak számít. A mély tanulás egyik legfontosabb alkalmazása a jelenetértelmezésben (scene understanding) történik, ahol az környezetben található objektumok minél több jellemzőjét és ezek kapcsolatait kívánjuk feltárni.

Az utóbbi néhány évben a mély tanulás kutatásának egyik fontos fókusza a különböző mobilis robotokban történő alkalmazása, amelyek a robot aktuális környezetének átfogó ismeretét igénylik. Az alkalmazások közül kiemelhető az autonóm járművek területe, ahol az objektumokról meghatározott információ teljessége és megbízhatósága kiemelten fontos.

A diplomatervezés során a hallgató feladata egy olyan algoritmus készítése, amely képes bemeneti kép vagy videofolyam alapján az abban található objektumok felismerésére és a jármű számára releváns információk (sebesség, forma, relációk stb.) kinyerésére.

A hallgató feladatának a következőkre kell kiterjednie:

- Tanulmányozza át a téma releváns szakirodalmát. Vizsgálja meg, hogy más műhelyek milyen megoldásokat alkalmaznak.
- Készítsen rendszertervet egy megoldásra, amely képes a jelenetértelmezés elvégzésére.
- Végezze el az algoritmus fejlesztését és tanítását.
- Tesztelje a megoldás pontosságát és hatékonyságát, valamint végezze el a tanuló algoritmus validációját.
- Vizsgálja a megoldást valósidejűség szempontjából.

Tanszéki konzulens: Szemenyei Márton



Budapest University of Technology and Economics

Faculty of Electrical Engineering and Informatics
Department of Control Engineering and Information Technology

Driving Scene Understanding in Simulation with Stereo RGB imaging and CNN synergy

MASTER'S THESIS

Author

Najib Ghadri

Advisor

Márton Szemenyei

May 29, 2020

Contents

Abstract	i
1 Introduction	1
1.1 Proposed solution	3
1.2 Summary of results	5
1.3 Thesis structure	6
2 Related work	7
3 Kind of perceptions	8
4 Assumptions made and limitations	9
5 Design and implementation	10
6 Results	11
7 Experimental results	12
8 Improvement notes	13
9 Conclusion	14
Acknowledgements	15
10 L^AT_EX-eszközök	16
10.1 A szerkesztéshez használatos eszközök	16
10.2 A dokumentum lefordítása Windows alatt	17
10.3 Eszközök Linuxhoz	17
11 A dolgozat formai kivitele	19
11.1 A dolgozat kimérete	19
11.2 A dolgozat nyelve	19

11.3 A dokumentum nyomdatechnikai kivitele	19
12 A L^AT_EX-sablon használata	21
12.1 Címkék és hivatkozások	21
12.2 Ábrák és táblázatok	21
12.3 Felsorolások és listák	23
12.4 Képletek	24
12.5 Irodalmi hivatkozások	25
12.6 A dolgozat szerkezete és a forrásfájlok	28
12.7 Alapadatok megadása	30
12.8 Új fejezet írása	30
12.9 Definíciók, tételek, példák	30
Bibliography	31
Appendix	32

HALLGATÓI NYILATKOZAT

Alulírott *Ghadri Najib*, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2020. május 29.

Ghadri Najib
hallgató

Abstract

Autonomous driving is undoubtedly the future of transportation. The comfort that it brings us is what drives us to work on making it real. We already have autonomous systems in public transportation in abundance, but it is different when we talk about the car roads. Driving a car requires near-human intelligence due to the nature of the environment, in fact it is impossible to define the environment. A train's or subways's environment can be defined mathematically and hence controlled easily, but for a machine to drive a car, it has to understand what we understand, and what we understand is even hard to define ourselves.

Computer science has come a long way, and we have already seen the rise of artificial intelligence algorithms and their effectiveness. Out of these methods Deep Learning and Convolutional Neural Networks are key tools in achieving our goal. With these algorithms computers learn general concepts of the world, and this is essential to make a safe autonomous driving (AD) system. We will see in this work briefly what they are and how they work.

Some notable companies have already achieved a high level of AD, most notably Tesla, and another AD supplier MobilEye. These companies use algorithms that are developed globally and publicly and I am going to use the same algorithms to partly achieve what they have achieved.

In this work I am going to create a Scene Understanding system specialized for driving situations. I choose to evaluate the system on a virtual car driving simulation called CARLA Sim, that is going to benefit us to measure our rate of success.

I researched how existing autonomous driving systems have been built, and inspired by them I designed a system that is capable of recognizing important information for a car on the road. I used stereo imaging of multiple RGB cameras mounted on top of our virtual car for depth estimation and used trained Convolutional Neural Networks to then perform further infomration extraction from the images and perform detection for each frame of the simulation. I made a 3D webvisualizer that is able to show us the difference between ground truth information extracted programatically from the simulator and the detection infomration while simultaneously play a montage video of the simulation. Finally I evaluated the system and measured it's validity for real situations and provided further improvement notes on my work. This thesis is also published on <https://najibghadri.com/msc-thesis/> where you can try the 3D webvisualizer.

Chapter 1

Introduction

I am very passionate about artificial intelligence and as much inspired by the work of tech companies such as Tesla. Tesla has managed create cutting edge technology, creating compelling and practical electric cars combined with their Tesla Autopilot system. It has become iconic to sit in a Tesla and watch it drive itself. Tesla has already driven 3 billion drives on autopilot, their access to data is most likely number one in the world. There are other important companies who develop autopilot systems, one of them is MobilEye an Israeli subsidiary of Intel corporation that was actually a supplier of Tesla until they set apart in part due to disagreements on how the technology should be built, which is an important topic that I will talk about.

There are a couple of topics we should establish first. The first being levels of autopilot systems as defined by SAE (Society of Automotive Engineers) (figure 1.1).

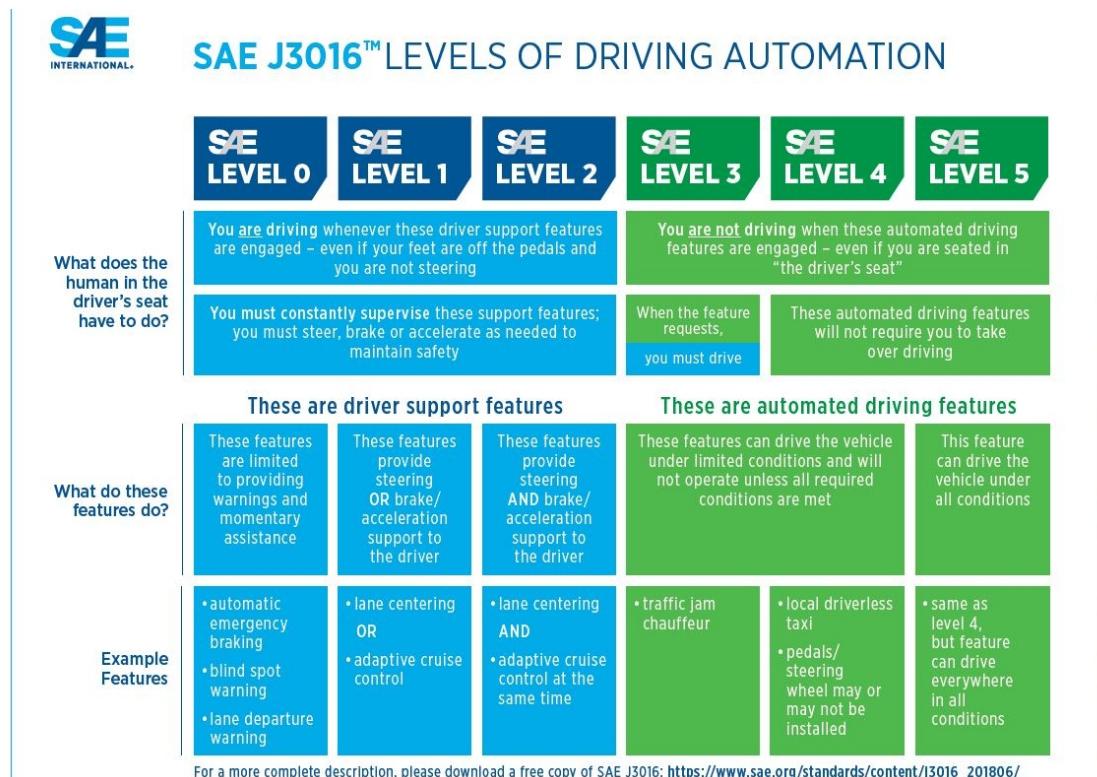


Figure 1.1: Levels of driving automation defined in SAE J3016 [1]

From level 0 to 2 are automations where the human is still required to fully monitor the driving environment. Tesla's autopilot is Level 2 which is partial automation that includes control of steering and both acceleration and deceleration. From Level 3 the human is not required to monitor the environment. Full automation, where the driver is not expected to intervene and the vehicle is able to handle all situations is on Level 5. In order to achieve that level the autopilot must fully understand the environment.

This is however very difficult. The algorithms that we know today are not enough to achieve understanding of the environment yet. Even Convolutional Neural Networks (CNNs) are not capable of understanding deep concepts of the world. CNNs are mainly used in computer vision and are useful when we want to recognize patterns that appear anywhere in 2D images. Today we are able to classify images, detect and localize objects, segment images to very high accuracy, however this doesn't mean the computer *understands* the scenes. Furthermore these algorithms are trained very specifically: To build a detection neural network (NN) first a meticulous dataset must be created that tells the algorithm what must be detected - we call this the ground truth, or training data set. Then the NN must be trained and optimized until it yields a low error on the test dataset. We call this Deep Learning due to the fact that the networks contain millions of parameters that are trained through hundreds of thousands of iterations. This is not close to what might be general AI.

In this sense we can argue about the meaning of "scene understanding". There is research going on in that direction most notably in my opinion by Yann LeCun the director of Facebook AI and professor at NYU, who works a concept called energy-based models. The Energy-based model that is a form of generative model allow a mathematical "bounding" or "learning" of a data distribution in any dimension. Upon prediction the model tries to generate a possible future for the current model in time, where the generated future model acts as the predictor itself. Generative adversarial networks are a type of these models. This is in contrast to the other main machine learning approach that is the discriminative model which is what we use mostly. Perceptrons such as NNs and CNNs, support vector machines fall into this category, however the distinction is still not clear.

For the purpose of this thesis hence it is important to define what a system capable of understanding scenes in driving situations means. The essentials are the following:

- Lane and path detection
- Driveable area detection
- Object detection: cars, pedestrians, etc.
- Object localization in 3D real world space
- Object tracking and identification
- Foreign object detection: anything that shouldn't be where it is
- Traffic light and sign understanding
- Handling occlusion of objects
- Pedestrian crossing detection
- Knowledge of surroundings and road for example with the help of high definition maps

In an ideal world, where all cars are autonomous these perceptions would be enough, however the future of self-driving cars is going to be a transition, where both humans and machines will drive on the roads. We humans already account for each other (we try as we can), but self-driving cars will have to account for us too. We might not be smart but driving on the road sometimes requires improvisation to save a situation and we might need a more general AI.

For the vehicle to understand it's surroundings first of all it needs sensors. Each company goes differently about the sensor suite, and it is quite interesting to examine each solution. I will talk about this in the next chapter, Related works.

1.1 Proposed solution

As we said to develop our system first of all we need data. There are a lot of datasets available on the internet for car driving. They include object detections, segmentations, map data, lidar data. Some of the most notable ones are the nuScene dataset¹, Waymo dataset² from Google's self-driving car company or the Cityscapes dataset³ and more. Each of these datasets are very good, however they are not really helpful for our case.

In order to localize object we will need to localize them, and for that I use stereo imaging. Each AD system today employs stereo camera setting because it is a very simple and quite accurate way of estimating depth for each pixel in an image. In order to have the *freedom* to create a custom camera setting I cannot rely on these datasets. Furthermore, I want to measure the success rate of my detector however there is no dataset that contains all the necessary information, because in fact it is not possible to collect everything from the real world.

This is why I choose to use a *simulation* instead to test the system. Using a simulation gives me a huge amount of freedom. My research work started in looking for simulations that let me extract data from the simulation in each frame and let's me create custom sensor settings. The first thing that came to my mind was GTA V the well-known free-world game, but soon I found out it isn't going to be useful. I found that there are dedicated projects for this, and a couple of companies have already invested in this. There is Deepdrive from Voyage auto⁴, an American AD supplier, NVIDIA has a project going on called Drive Constellation⁵ and another project called RFPro⁶. However these are either not opensource or not mature enough. The best that I found and ended up basing my project on is called CARLA Simulator⁷ [2] that is a quite mature and fully opensource project developed by the Barcelonian university UAB's computer vision CVC Lab supported by Intel, Toyota, GM and others. This simulator has a really good API that fulfills my requirements.

The sensor suite includes 8 RGB cameras mounted on the roof of the car as shown on figure 1.3. As the title of the thesis says, I only used RGB cameras and no other sensors. This is the same approach Tesla is taking, contrary to almost all other players in the field including MobilEye and Waymo. Lidar data is good for correction, but it is better if the

¹nuScene dataset <https://www.nuscenes.org/>

²Waymo dataset <https://waymo.com/open/>

³Cityscapes dataset <https://www.cityscapes-dataset.com/>

⁴Deepdrive Voyage <https://deepdrive.voyage.auto/>

⁵NVIDIA Drive Constellation <https://www.nvidia.com/en-us/self-driving-cars/drive-constellation/>

⁶RFPro <http://www.rfpro.com/>

⁷CARLA Sim <http://carla.org/>



Figure 1.2: A screenshot from CARLA

AI can equally perform using only RGB cameras since it is a more general solution that is closer to how we humans perceive the environment.



Figure 1.3: How the cameras are set up on the roof

The system uses state-of-the-art detection, localization and segmentation model Detectron2 [3] a MASK R-CNN conv net model based on Residual neural networks and Feature Pyramid Networks trained on the COCO general dataset⁸.

Finally I develop a 3D webvisualizer that lets us replay the ground truth and detection log simultaneously and compare the error between the two.

figure 1.4 depicts this taskflow.

⁸COCO dataset <http://cocodataset.org/>

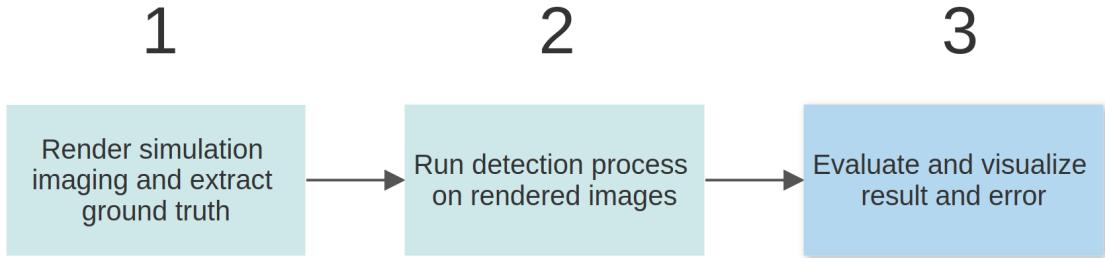


Figure 1.4: Task flow

1.2 Summary of results

The result is a detector that is capable of localizing vehicles, and pedestrians on the road up to 100 meters with an accuracy of 50cm in an angle of 270° centered to the front. The virtual vehicle has 8 RGB cameras mounted creating 4 stereo sides. The algorithm is written in Python and uses PyTorch, with that on an NVIDIA Titan X GPU the detector can perform in 2.7FPS for one side, ie. for two cameras. In an embedded optimized system using C or C++ code this can easily be improved to even 60FPS creating a real-time system. The code cannot perform lane detection yet, but that would have been the easier part. The webvisualizer let's us replay the simulation frame by frame and see the detection error for each actor in the scene. It also shows a montage the original, detection and depthmap. Below figure 1.5 shows a screenshot of the webvisualizer in action.

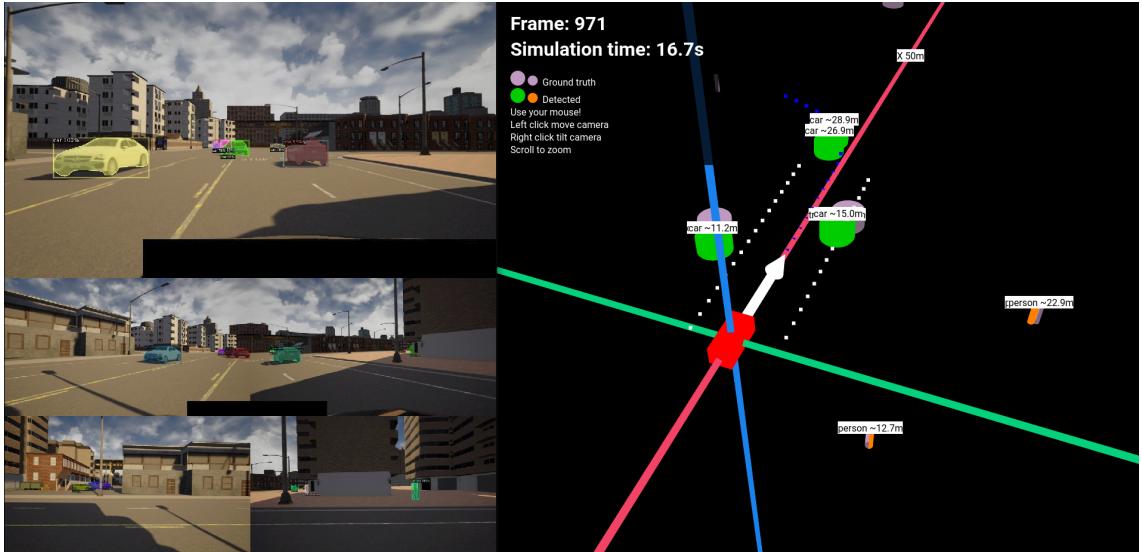


Figure 1.5: 3D wevisualizer

All of the code for the thesis, detector, simulator configuration and webvisualizer is available on <https://github.com/najibghadri/msc-thesis> and you can access the web-visualizer and interactively replay and test simulations on <https://najibghadri.com/msc-thesis/>.

1.3 Thesis structure

In the next chapter, Chapter 2 I analyze and compare two self-driving solutions: Tesla autopilot and MobilEye autopilot.

In Chapter 3 I talk about different kind of perceptions, state-of-the-art Convolutional Networks and computer vision algorithms that are useful for our use-case. In Chapter 4 I define the technical assumptions that I made in order to simplify the task and the resulting limitations. In Chapter 5 I detail the design and implementation of the simulator configuration, the detector algorithm and the webvisualizer.

Then in Chapter 6 I present different measurements and results, and in Chapter 7 I present experimentations that ended up not being part of the detection. Finally I discuss ways to improve the system in Chapter 8

- Structure of Thesis - Each chapter - All code and thesis available at <https://github.com/najibghadri/msc-thesis> and the published version on my website

Chapter 2

Related work

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

Chapter 3

Kind of perceptions

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

Chapter 4

Assumptions made and limitations

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

Chapter 5

Design and implementation

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

Chapter 6

Results

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

Chapter 7

Experimental results

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

Chapter 8

Improvement notes

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

Chapter 9

Conclusion

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

Acknowledgements

Ez nem kötelező, akár törölhető is. Ha a szerző szükségét érzi, itt lehet köszönetet nyilvánítani azoknak, akik hozzájárultak munkájukkal ahhoz, hogy a hallgató a szakdolgozatban vagy diplomamunkában leírt feladatokat sikeresen elvégezze. A konzulensnek való köszönetnyilvánítás sem kötelező, a konzulensnek hivatalosan is dolga, hogy a hallgatót konzultálja.

Chapter 10

L^AT_EX-eszközök

10.1 A szerkesztéshez használatos eszközök

Ez a sablon TeXstudio 2.8.8 szerkesztővel készült. A TeXstudio egy platformfüggetlen, Windows, Linux és Mac OS alatt is elérhető L^AT_EX-szerkesztőprogram számtalan hasznos szolgáltatással (figure 10.1). A szoftver ingyenesen letölthető¹.

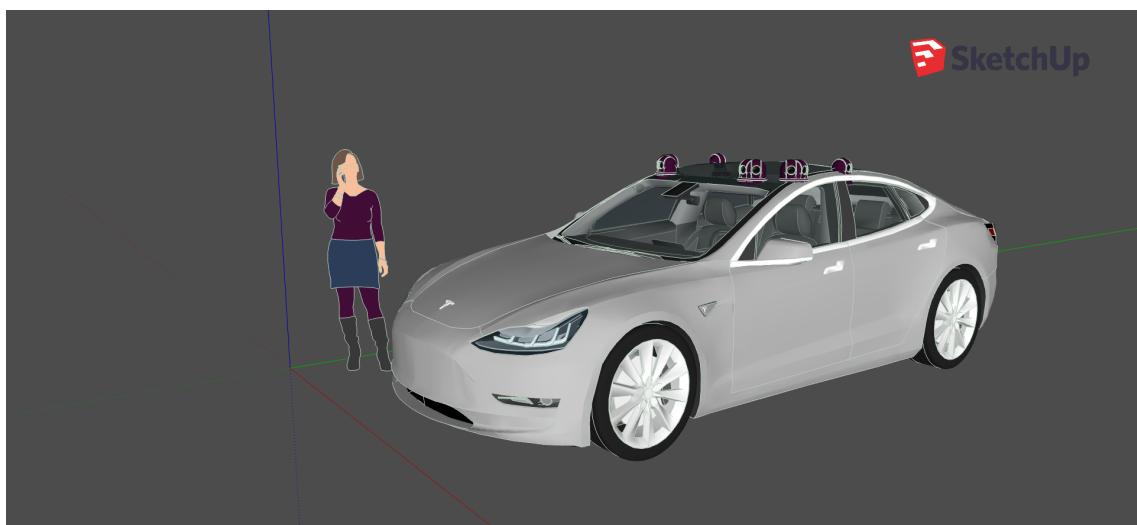


Figure 10.1: A TeXstudio L^AT_EX-szerkesztő.

A TeXstudio telepítése után érdemes még letölteni a magyar nyelvű helyesírásellenőrzőszótárakat hozzá. A TeXstudio az OpenOffice-hoz használatos formátumot tudja kezelní. A TeXstudio beállításainál a General fülön a Dictionaries résznél tudjuk megadni, hogy melyik szótárat használja.

Egy másik használható Windows alapú szerkesztőprogram a LEd² (LaTeX Editor), a TeXstudio azonban stabilabb, gyorsabb, és jobban használható.

¹A TeXstudio hivatalos oldala: <http://texstudio.sourceforge.net/>

²A LEd hivatalos oldala: <http://www.latexeditor.org/>

10.2 A dokumentum lefordítása Windows alatt

A TeXstudio és a LEd kizárolag szerkesztőprogram (bár az utóbbiban DVI-nézegető is van), így a dokumentum fordításához szükséges eszközöket nem tartalmazza. Windows alatt alapvetően két lehetőség közül érdemes választani: MiKTeX (<http://miktex.org/>) és TeX Live (<http://www.tug.org/texlive/>) programcsomag. Az utóbbi működik Mac OS X, GNU/Linux alatt és Unix-származékokon is. A MiKTeX egy alapcsomag telepítése után mindenki letölthető a használt funkcióhoz szükséges, de lokálisan hiányzó \TeX -csomagokat, míg a TeX Live DVD ISO verzóban férhető hozzá. Ez a dokumentum TeX Live 2008 programcsomag segítségével fordult, amelynek DVD ISO verziója a megadott oldalról letölthető. A sablon lefordításához a disztribúcióban szereplő magyar.ldf fájlt a <http://www.math.bme.hu/latex/> változatra kell cserélni, vagy az utóbbi változatot be kell másolni a projekt-könyvtárba (ahogy ezt meg is tettük a sablonban) különben anomáliák tapasztalhatók a dokumentumban (pl. az ábra- és táblázat-aláírások formátuma nem a beállított lesz, vagy bizonyos oldalakon megjelenik alapértelmezésben egy fejléc). A TeX Live 2008-at még nem kell külön telepíteni a gépre, elegendő DVD-ről (vagy az ISO fájlból közvetlenül, pl. DaemonTools-szal) használni.

Ha a MiKTeX csomagot használjuk, akkor parancssorból a következő módon tudjuk újrafordítani a teljes dokumentumot:

```
$ texify -p thesis.tex
```

A `texify` parancs a MiKTeX programcsomag `miktex/bin` alkönyvtárában található. A parancs gondoskodik arról, hogy a szükséges lépéseket (fordítás, hivatkozások generálása stb.) a megfelelő sorrendben elvégezze. A `-p` kapcsoló hatására PDF-et generál. A fordítást és az ideiglenes fájlok törlését elvégezhetjük a sablonhoz mellékelt `manual_build.bat` szkript segítségével is.

A \TeX -eszközöket tartalmazó programcsomag binárisainak elérési útját gyakran be kell állítani a szerkesztőprogramban, például TeXstudio esetén legegyszerűbben az Options / Configure TeXstudio... / Commands menüponttal előhívott dialógusablakban tehetjük ezt meg.

A PDF-L^AT_EX használata esetén a generált dokumentum közvetlenül PDF-formátumban áll rendelkezésre. Amennyiben a PDF-fájl egy PDF-nézőben (pl. Adobe Acrobat Reader vagy Foxit PDF Reader) meg van nyitva, akkor a fájlleírót a PDF-néző program tipikusan lefoglalja. Ilyen esetben a dokumentum újrafordítása hibaüzenettel kilép. Ha bezárjuk és újra megnyitjuk a PDF dokumentumot, akkor pedig a PDF-nézők többsége az első oldalon nyitja meg a dokumentumot, nem a legutóbb olvasott oldalon. Ezzel szemben például az egyszerű és ingyenes [Sumatra PDF](#) nevű program képes arra, hogy a megnyitott dokumentum megváltozását detektálja, és frissítse a nézetet az aktuális oldal megtartásával.

10.3 Eszközök Linuxhoz

Linux operációs rendszer alatt is rengeteg szerkesztőprogram van, pl. a KDE alapú Kile jól használható. Ez ingyenesen letölthető, vagy éppenséggel az adott Linux-disztribúció eleve tartalmazza, ahogyan a dokumentum fordításához szükséges csomagokat is. Az Ubuntu Linux disztribúciók alatt például legtöbbször a `texlive-*` csomagok telepítésével használhatók a L^AT_EX-eszközök. A jelen sablon fordításához szükséges csomagok (kb. 0,5 GB) az alábbi parancssal telepíthetők:

```
$ sudo apt-get install texlive-latex-extra texlive-fonts-extra texlive-fonts-recommended  
texlive-xetex texlive-science
```

Amennyiben egy újabb csomag hozzáadása után hiányzó fájlra utaló hibát kapunk a fordítótól, telepítenünk kell az azt tartalmazó TeX Live csomagot. Ha pl. a `bibentry` csomagot szeretnénk használni, futtassuk az alábbi parancsot:

```
$ apt-cache search bibentry  
texlive-luatex - TeX Live: LuaTeX packages
```

Majd telepítsük fel a megfelelő TeX Live csomagot, jelen esetben a ‘texlive-lualatex’-et. (Egy LaTeX csomag több TeX Live csomagban is szerepelhet.)

Ha gyakran szerkesztünk más L^AT_EX dokumentumokat is, kényelmes és biztos megoldás a teljes TeX Live disztribúció telepítése, ez azonban kb. 4 GB helyet igényel.

```
sudo apt-get install texlive-full
```

Chapter 11

A dolgozat formai kivitele

Az itt található információk egy része a BME VIK Hallgatói Képviselet által készített „Utolsó félév a villanykaron” c. munkából lett kis változtatásokkal átemelve. Az eredeti dokumentum az alábbi linken érhető el: <http://vik.hk/hirek/diplomafelev-howto-2015>.

11.1 A dolgozat kimérete

Szakdolgozat esetében minimum 30, 45 körüli ajánlott oldalszám lehet az iránymutató. De mindenkor érdemes rágérdezni a konzulensnél is az elvárásokra, mert tanszékenként változóak lehetnek az elvárások.

Mesterképzésen a Diplomatervezés 1 esetében a beszámoló még inkább az Önálló laboratóriumi beszámolóhoz hasonlít, tanszékenként eltérő formai követelményekkel, – egy legalább 30 oldal körüli dolgozat az elvárt – és az elmúlt fél éves munkáról szól. De egyben célszerű, ha ez a végleges diplomaterv alapja is. (A végleges 60-90 oldal körülbelül a hasznos részre nézve)

11.2 A dolgozat nyelve

Mivel Magyarországon a hivatalos nyelv a magyar, ezért alapértelmezésben magyarul kell megírni a dolgozatot. Aki külföldi posztgraduális képzésben akar részt venni, nemzetközi szintű tudományos kutatást szeretne végezni, vagy multinacionális cégnél akar elhelyezkedni, annak célszerű angolul megírnia diplomadolgozatát. Mielőtt a hallgató az angol nyelvű verzió mellett dönt, erősen ajánlott mérlegelni, hogy ez mennyi többletmunkát fog a hallgatónak jelenteni fogalmazás és nyelvhelyesség terén, valamint – nem utolsó sorban – hogy ez mennyi többletmunkát fog jelenteni a konzulens illetve bíráló számára. Egy nehezen olvasható, netalán érthetetlen szöveg teher minden játékos számára.

11.3 A dokumentum nyomdatechnikai kivitele

A dolgozatot A4-es fehér lapra nyomtatva, 2,5 centiméteres margóval (+1 cm kötésbeni), 11–12 pontos betűmérettel, talpas betűtípussal és másfeles sorközzel célszerű elkészíteni.

Annak érdekében, hogy a dolgozat külsőleg is igényes munka benyomását keltse, érdemes figyelni az alapvető tipográfiai szabályok betartására [?].

Chapter 12

A L^AT_EX-sablon használata

Ebben a fejezetben röviden, implicit módon bemutatjuk a sablon használatának módját, ami azt jelenti, hogy sablon használata ennek a dokumentumnak a forráskódját tanulmányozva válik teljesen világossá. Amennyiben a szoftver-kéretrendszer telepítve van, a sablon alkalmazása és a dolgozat szerkesztése L^AT_EX-ben a sablon segítségével tapasztalataink szerint jóval hatékonyabb, mint egy WYSWYG (*What You See is What You Get*) típusú szövegszerkesztő esetén (pl. Microsoft Word, OpenOffice).

12.1 Címkek és hivatkozások

A L^AT_EX dokumentumban címkeket (`\label`) rendelhetünk ábrákhoz, táblázatokhoz, fejezetekhez, listákhoz, képletekhez stb. Ezekre a dokumentum bármely részében hivatkozhatunk, a hivatkozások automatikusan feloldásra kerülnek.

A sablonban makrókat definiáltunk a hivatkozások megkönnyítéséhez. Ennek megfelelően minden ábra (*figure*) címkeje `fig:` kulcsszóval kezdődik, míg minden táblázat (*table*), képlet (*equation*), fejezet (*section*) és lista (*listing*) rendre a `tab:, eq:, sec:` és `lst:` kulcsszóval kezdődik, és a kulcsszavak után tetszőlegesen választott címke használható. Ha ezt a konvenciót betartjuk, akkor az előbbi objektumok számára rendre a `\figref`, `\tabref`, `\eqref`, `\sectref` és `\listref` makrókkal hivatkozhatunk. A makrók paramétere a címke, amelyre hivatkozunk (a kulcsszó nélkül). Az összes említett hivatkozástípus, beleértve az `\url` kulcsszóval bevezetett web-hivatkozásokat is a `hyperref`¹ csomagnak köszönhetően aktívak a legtöbb PDF-nézegetőben, rájuk kattintva a dokumentum megfelelő oldalára ugrik a PDF-néző vagy a megfelelő linket megnyitja az alapértelmezett böngészővel. A `hyperref` csomag a kimeneti PDF-dokumentumba könyvjelzőket is készít a tartalomjegyzékből. Ez egy szintén aktív tartalomjegyzék, amelynek elemeire kattintva a nézegető behozza a kiválasztott fejezetet.

12.2 Ábrák és táblázatok

Használunk vektorgrafikus ábrákat, ha van rá módunk. PDFLaTeX használata esetén PDF formátumú ábrákat lehet beilleszteni könnyen, az EPS (PostScript) vektorgrafikus képformátum beillesztését a PDFLaTeX közvetlenül nem támogatja (de lehet konvertálni,

¹Segítségével a dokumentumban megjelenő hivatkozások nem csak dinamikussá válnak, de színezhetők is, bővebbet erről a csomag dokumentációjában találunk. Ez egyúttal egy példa lábjegyzet írására.

lásd később). Ha vektorgrafikus formában nem áll rendelkezésünkre az ábra, akkor a veszteségmentes PNG, valamint a veszteséges JPEG formátumban érdemes elmenteni. Figyeljünk arra, hogy ilyenkor a képek felbontása elég nagy legyen ahhoz, hogy nyomtatásban is megfelelő minőséget nyújtson (legalább 300 dpi javasolt). A dokumentumban felhasznált képfájlokat a dokumentum forrása mellett érdemes tartani, archiválni, mivel ezek hiányában a dokumentum nem fordul újra. Ha lehet, a vektorgrafikus képeket vektorgrafikus formátumban is érdemes elmenteni az újrafelhasználhatóság (az átszerkeszthetőség) érdekében.

Kapcsolási rajzok legtöbbször kimásolhatók egy vektorgrafikus programba (pl. CorelDraw) és onnan nagyobb felbontással raszterizálva kimenthetők PNG formátumban. Ugyanakkor kiválasztott ábrák készíthetők Microsoft Visio vagy hasonló program használatával is: Visio-ból az ábrák közvetlenül PDF-be is menthetők.

Lehetőségeink Matlab ábrák esetén:

- Képernyőlopás (*screenshot*) is elfogadható minőségű lehet a dokumentumban, de általában jobb felbontást is el lehet érni más módszerrel.
- A Matlab ábrát a *File/Save As* opcióval lementhetjük PNG formátumban (ugyanazt is érvényes, mint korábban, ezért nem javasoljuk).
- A Matlab ábrát az *Edit/Copy figure* opcióval kimásolhatjuk egy vektorgrafikus programba is és onnan nagyobb felbontással raszterizálva kimenthetjük PNG formátumban (nem javasolt).
- Javasolt megoldás: az ábrát a *File/Save As* opcióval EPS *vektorgrafikus* formátumban elmentjük, PDF-be konvertálva beillesztjük a dolgozatba.

Az EPS kép az *epstopdf* programmal² konvertálható PDF formátumba. Célszerű egy batch-fájlt készíteni az összes EPS ábra lefordítására az alábbi módon (ez Windows alatt működik).

```
@echo off
for %%j in (*.eps) do (
    echo converting file "%%j"
    epstopdf "%%j"
)
echo done .
```

Egy ilyen parancsfájlt (*convert.cmd*) elhelyeztük a sablon *figures\eps* könyvtárába, így a felhasználónak csak annyi a dolga, hogy a *figures\eps* könyvtárba kimenti az EPS formátumú vektorgrafikus képet, majd lefuttatja a *convert.cmd* parancsfájlt, ami PDF-be konvertálja az EPS fájlt.

Ezek után a PDF-ábrát ugyanúgy lehet a dokumentumba beilleszteni, mint a PNG-t vagy a JPEG-et. A megoldás előnye, hogy a lefordított dokumentumban is vektorgrafikusan tárolódik az ábra, így a mérete jóval kisebb, mintha raszterizáltuk volna beillesztés előtt. Ez a módszer minden – az EPS formátumot ismerő – vektorgrafikus program (pl. CorelDraw) esetén is használható.

A képek beillesztésére a chapter 10ben mutattunk be példát (figure 10.1). Az előző mondatban egyúttal az automatikusan feloldódó ábrahivatkozásra is láthatunk példát. Több képfájlt is beilleszthetünk egyetlen ábrába. Az egyes képek közötti horizontális és vertikális margót metrikusan szabályozhatjuk (figure 12.1). Az ábrák elhelyezését számtalan

²a korábban említett L^AT_EX-disztribúciókban megtalálható

tipográfiai szabály egyidejű teljesítésével a fordító maga végzi, a dokumentum írója csak preferenciáit jelezheti a fordító felé (olykor ez bosszúságot is okozhat, ilyenkor pl. a kép méretével lehet játszani).

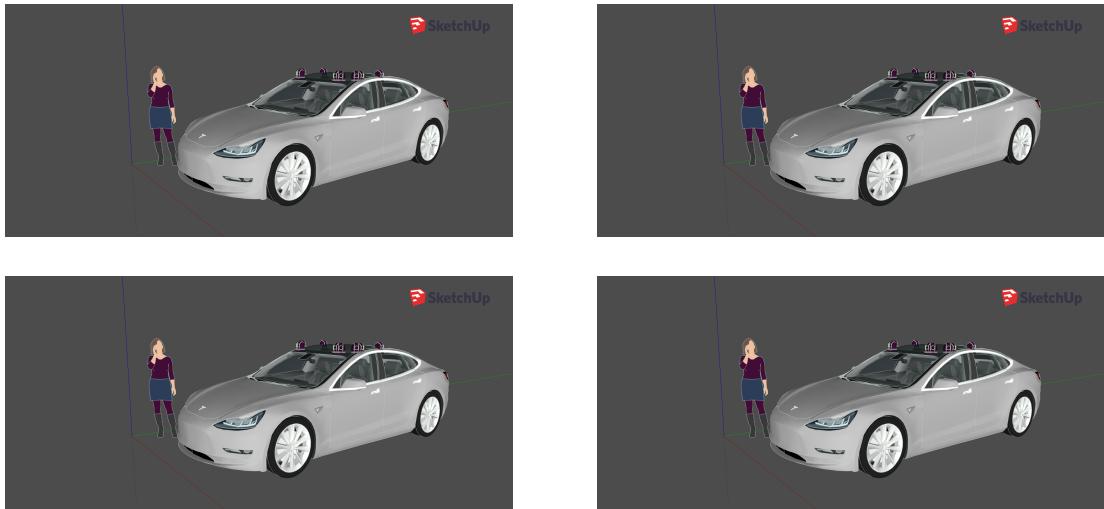


Figure 12.1: Több képfájl beillesztése esetén térközöket is érdemes használni.

A táblázatok használatára a 12.1 táblázat mutat példát. A táblázatok formázásához hasznos tanácsokat találunk a booktabs csomag dokumentációjában.

Órajel	Frekvencia	Cél pin
CLKA	100 MHz	FPGA CLK0
CLKB	48 MHz	FPGA CLK1
CLKC	20 MHz	Processzor
CLKD	25 MHz	Ethernet chip
CLKE	72 MHz	FPGA CLK2
XBUF	20 MHz	FPGA CLK3

Table 12.1: Az órajel-generátor chip órajel-kimenetei.

12.3 Felsorolások és listák

Számozatlan felsorolásra mutat példát a jelenlegi bekezdés:

- *első bajusz*: ide lehetne írni az első elem kifejését,
- *második bajusz*: ide lehetne írni a második elem kifejését,
- *ez meg egy szakáll*: ide lehetne írni a harmadik elem kifejését.

Számozott felsorolást is készíthetünk az alábbi módon:

1. *első bajusz*: ide lehetne írni az első elem kifejését, és ez a kifejtés így néz ki, ha több sorosra sikeredik,
2. *második bajusz*: ide lehetne írni a második elem kifejését,

3. *ez meg egy szakáll*: ide lehetne írni a harmadik elem kifejését.

A felsorolásokban sorok végén vessző, az utolsó sor végén pedig pont a szokásos írásjel. Ez alól kivételt képezhet, ha az egyes elemek több teljes mondatot tartalmaznak.

Listákban a dolgozat szövegétől elkülönítendő kódrészleteket, programsorokat, pszeudo-kódokat jeleníthetünk meg (12.1. kódrészlet).

```
\begin{enumerate}
    \item \emph{első bajusz:} ide lehetne írni az első elem kifejését,
        és ez a kifejtés így néz ki, ha több sorosra sikeredik,
    \item \emph{második bajusz:} ide lehetne írni a második elem kifejését,
    \item \emph{ez meg egy szakáll:} ide lehetne írni a harmadik elem kifejését.
\end{enumerate}
```

Listing 12.1: A fenti számoszott felsorolás L^AT_EX-forráskódja

A lista keretét, háttérszínét, egész stílusát megválaszthatjuk. Ráadásul különféle programnyelveket és a nyelveken belül kulcsszavakat is definiálhatunk, ha szükséges. Erről bővebbet a `listings` csomag hivatalos leírásában találhatunk.

12.4 Képletek

Ha egy formula nem túlságosan hosszú, és nem akarjuk hivatkozni a szövegből, mint például a $e^{i\pi} + 1 = 0$ képlet, *szövegközi képletek*nt szokás leírni. Csak, hogy másik példát is lássunk, az $U_i = -d\Phi/dt$ Faraday-törvény a $\text{rot } E = -\frac{dB}{dt}$ differenciális alakban adott Maxwell-egyenlet felületre vett integráljából vezethető le. Látható, hogy a L^AT_EX-fordító a sorközöket betartja, így a szöveg szedése esztétikus marad szövegközi képletek használata esetén is.

Képletek esetén az általános konvenció, hogy a kis felkívér betűk (**v**) oszlopvektort – és ennek megfelelően **v**^T sorvektort – a kapitális felkívér betűk (**V**) mátrixot jelölnek. Ha ettől el szeretnénk térni, akkor az alkalmazni kívánt jelölésmódot célszerű külön alfejezetben definiálni. Ennek megfelelően, amennyiben **y** jelöli a mérések vektorát, ϑ a paraméterek vektorát és $\hat{\mathbf{y}} = \mathbf{X}\vartheta$ a paraméterekben lineáris modellt, akkor a *Least-Squares* értelemben optimális paraméterbecslő $\hat{\vartheta}_{LS} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$ lesz.

Emellett kiemelt, sorszámoszott képleteket is megadhatunk, ennél az `equation` és a `eqnarray` környezetek helyett a korszerűbb `align` környezet alkalmazását javasoljuk (több okból, különféle problémák elkerülése végett, amelyekre most nem térünk ki). Tehát

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \quad (12.1)$$

$$\mathbf{y} = \mathbf{C}\mathbf{x}, \quad (12.2)$$

ahol **x** az állapotvektor, **y** a mérések vektora és **A**, **B** és **C** a rendszert leíró paramétermátrixok. Figyeljük meg, hogy a két egyenletben az egyenlőségjelek egymáshoz igazítva jelennek meg, mivel a minden két karakter előzi meg a kódban. Lehetőség van számoszthatlan kiemelt képlet használatára is, például

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u},$$

$$\mathbf{y} = \mathbf{C}\mathbf{x}.$$

Mátrixok felírására az $\mathbf{Ax} = \mathbf{b}$ inhomogén lineáris egyenlet részletes kifejtésével mutatunk példát:

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}. \quad (12.3)$$

A `\frac` utasítás hatékonyságát egy általános másodfokú tag átviteli függvényén keresztül mutatjuk be, azaz

$$W(s) = \frac{A}{1 + 2T\xi s + s^2 T^2}. \quad (12.4)$$

A matematikai mód minden szimbólumának és képességének a bemutatására természeten itt nincs lehetőség, de gyors referenciaként hatékonyan használhatók a következő linkek:
http://www.artofproblemsolving.com/LaTeX/AoPS_L_GuideSym.php,
<http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf>,
<ftp://ftp.ams.org/pub/tex/doc/amsmath/short-math-guide.pdf>.

Ez pedig itt egy magyarázat, hogy miért érdemes align környezetet használni:
<http://texblog.net/latex-archive/math/eqnarray-align-environment/>.

12.5 Irodalmi hivatkozások

Egy L^AT_EX dokumentumban az irodalmi hivatkozások definíciójának két módja van. Az egyik a `\thebibliography` környezet használata a dokumentum végén, az `\end{document}` lezárás előtt.

```
\begin{thebibliography}{9}
\bibitem{Lamport94} Leslie Lamport, \emph{\LaTeX: A Document Preparation System}. Addison Wesley, Massachusetts, 2nd Edition, 1994.
\end{thebibliography}
```

Ezek után a dokumentumban a `\cite{Lamport94}` utasítással hivatkozhatunk a forrásra. A fenti megadás viszonylag kötetlen, a szerző maga formázza az irodalomjegyzéket (ami gyakran inkonzisztens eredményhez vezet).

Egy sokkal professzionálisabb módszer a BiBT_EX használata, ezért ez a sablon is ezt támogatja. Ebben az esetben egy külön szöveges adatbázisban definiáljuk a forrásmunkákat, és egy külön stílusfájl határozza meg az irodalomjegyzék kinézetét. Ez, összhangban azzal, hogy külön formátumkonvenció határozza meg a folyóirat-, a könyv-, a konferenciacikk-stb. hivatkozások kinézetét az irodalomjegyzékben (a sablon használata esetén ezzel nem is kell foglalkoznia a hallgatónak, de az eredményt célszerű ellenőrizni). Felhasznált hivatkozások adatbázisa egy .bib kiterjesztésű szöveges fájl, amelynek szerkezetét a A 12.2 kód részlet demonstrálja. A forrásmunkák bevitelekor a sor végi vesszők külön figyelmet igényelnek, mert hiányuk a BiBT_EX-fordító hibaüzenetét eredményezi. A forrásmunkákat típus szerinti kulcsszó vezeti be (@book könyv, @inproceedings konferenciakiadványban megjelent cikk, @article folyóiratban megjelent cikk, @techreport valamelyik egyetem gondozásában megjelent műszaki tanulmány, @manual műszaki dokumentáció esetén stb.). Nemcsak a megjelenés stílusa, de a kötelezően megadandó mezők is típusról-típusra változnak. Egy jól használható referencia a <http://en.wikipedia.org/wiki/BibTeX> oldalon található.

```

@book{Wettl04,
  author      = {Ferenc Wettl and Gyula Mayer and Péter Szabó},
  publisher   = {Panem Könyvkiadó},
  title       = {\LaTeX-kézikönyv},
  year        = {2004},
}

@article{Candy86,
  author      = {James C. Candy},
  journaltitle = {{IEEE} Trans.\ on Communications},
  month       = {01},
  note        = {\doi{10.1109/TCOM.1986.1096432}},
  number      = {1},
  pages       = {72--76},
  title       = {Decimation for Sigma Delta Modulation},
  volume      = {34},
  year        = {1986},
}

@inproceedings{Lee87,
  author      = {Wai L. Lee and Charles G. Sodini},
  booktitle   = {Proc.\ of the IEEE International Symposium on Circuits and Systems},
  location    = {Philadelphia, PA, USA},
  month       = {05-4--7},
  pages       = {459--462},
  title       = {A Topology for Higher Order Interpolative Coders},
  vol         = {2},
  year        = {1987},
}

@thesis{KissPhD,
  author      = {Peter Kiss},
  institution = {Technical University of Timi\c{s}oara, Romania},
  month       = {04},
  title       = {Adaptive Digital Compensation of Analog Circuit Imperfections for Cascaded Delta-Sigma Analog-to-Digital Converters},
  type        = {phdthesis},
  year        = {2000},
}

@manual{Schreier00,
  author      = {Richard Schreier},
  month       = {01},
  note        = {\url{http://www.mathworks.com/matlabcentral/fileexchange/}},
  organization = {Oregon State University},
  title       = {The Delta-Sigma Toolbox v5.2},
  year        = {2000},
}

@misc{DipPortal,
  author      = {{Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Kar}},
  howpublished = {\url{http://diplomaterv.vik.bme.hu/}},
  title       = {Diplomaterv portál (2011. február 26.)},
}

@incollection{Mkrtychev:1997,
  author      = {Mkrtychev, Alexey},
  booktitle   = {Logical Foundations of Computer Science},
  doi         = {10.1007/3-540-63045-7_27},
  editor      = {Adian, Sergei and Nerode, Anil},
  isbn        = {978-3-540-63045-6},
  pages       = {266-275},
  publisher   = {Springer Berlin Heidelberg},
  series      = {Lecture Notes in Computer Science},
  title       = {Models for the logic of proofs},
  url         = {http://dx.doi.org/10.1007/3-540-63045-7_27},
  volume      = {1234},
  year        = {1997},
}

```

Listing 12.2: Példa szöveges irodalomjegyzék-adatbázisra BibTeX használata esetén.

A stílusfájl egy .sty kiterjesztésű fájl, de ezzel lényegében nem kell foglalkozni, mert vannak beépített stílusok, amelyek jól használhatók. Ez a sablon a BiBTEX-et használja, a hozzá tartozó adatbázisfájl a mybib.bib fájl. Megfigyelhető, hogy az irodalomjegyzéket a dokumentum végére (a \end{document} utasítás elő) beillesztett \bibliography{mybib} utasítással hozhatjuk létre, a stílusát pedig ugyanitt a \bibliographystyle{plain} utasítással adhatjuk meg. Ebben az esetben a plain előre definiált stílust használjuk (a sablonban is ezt állítottuk be). A plain stíluson kívül természetesen számtalan más előre definiált stílus is létezik. Mivel a .bib adatbázisban ezeket megadtuk, a BiBTEX-fordító is meg tudja különböztetni a szerzőt a címtől és a kiadótól, és ez alapján automatikusan generálódik az irodalomjegyzék a stílusfájl által meghatározott stílusban.

Az egyes forrásmunkákra a szövegből továbbra is a \cite paranccsal tudunk hivatkozni, így a 12.2. kódrészlet esetén a hivatkozások rendre \cite{Wett104}, \cite{Candy86}, \cite{Lee87}, \cite{KissPhD}, \cite{Schreirer00}, \cite{Mkrtychev:1997} és \cite{DipPortal}. Az egyes forrásmunkák sorszáma az irodalomjegyzék bővítésekor változhat. Amennyiben az aktuális számhoz illeszkedő névelőt szeretnénk használni, használjuk az \acite{} parancsot.

Az irodalomjegyzékben alapértelmezésben csak azok a forrásmunkák jelennek meg, amelyekre található hivatkozás a szövegben, és ez így alapvetően helyes is, hiszen olyan forrásmunkákat nem illik az irodalomjegyzékbe írni, amelyekre nincs hivatkozás.

Mivel a fordítási folyamat során több lépésekben oldódnak fel a szimbólumok, ezért gyakran többször is le kell fordítani a dokumentumot. Ilyenkor ez első 1-2 fordítás esetleg szimbólum-feloldásra vonatkozó figyelmeztető üzenettel zárul. Ha hibaüzenettel zárul bármelyik fordítás, akkor nincs értelme megismételni, hanem a hibát kell megkeresni. A .bib fájl megváltoztatáskor sokszor nincs hatása a változtatásnak azonnal, mivel nem minden fut újra a BibTeX fordító. Ezért célszerű a változtatás után azt manuálisan is lefuttatni (TeXstudio esetén Tools/Bibliography).

Hogy a szövegebe ágyazott hivatkozások kinézetét demonstráljuk, itt most sorban meghivatkozzuk a [?], [?], [?], [?], [?] és a [?] ³ forrásmunkát, valamint a [?] weboldalt.

Megjegyzendő, hogy az ékezes magyar betűket is tartalmazó .bib fájl az inputenc csomaggal betöltött latin2 betűkészlet miatt fordítható. Ugyanez a .bib fájl hibaüzenettel fordul egy olyan dokumentumban, ami nem tartalmazza a \usepackage[latin2]{inputenc} sort. Speciális igény esetén az irodalmi adatbázis általánosabb érvényűvé tehető, ha az ékezes betűket speciális latex karakterekkel helyettesítjük a .bib fájlban, pl. á helyett \'{a}-t vagy ő helyett \H{o}-t írunk.

Irodalomhivatkozásokat célszerű először olyan szolgáltatásokban keresni, ahol jó minőségű bejegyzések találhatók (pl. ACM Digital Library,⁴ DBLP,⁵ IEEE Xplore,⁶ SpringerLink⁷) és csak ezek után használni kevésbé válogatott forrásokat (pl. Google Scholar⁸). A jó minőségű bejegyzéseket is érdemes megfelelően tisztítani.⁹ A sablon angol nyelvű változatában használt plainnat beállítás egyik sajátossága, hogy a cikkhez generált hivatkozás

³Informatikai témaiban gyakran hivatkozunk cikkeket a Springer LNCS valamely kötetéből, ez a hivatkozás erre mutat egy helyes példát.

⁴<https://dl.acm.org/>

⁵<http://dblp.uni-trier.de/>

⁶<http://ieeexplore.ieee.org/>

⁷<https://link.springer.com/>

⁸<http://scholar.google.com/>

⁹<https://github.com/FTSRG/cheat-sheets/wiki/BibTeX-Fixing-entries-from-common-sources>

a cikk DOI-ját és URL-jét is tartalmazza, ami gyakran duplikátumhoz vezet – érdemes tehát a DOI-kat tartalmazó URL mezőket törölni.

12.6 A dolgozat szerkezete és a forrásfájlok

A diplomatervsablonban a TeX fájlok két alkönyvtárban helyezkednek el. Az `include` könyvtárban azok szerepelnek, amiket tipikusan nem kell szerkesztenünk, ezek a sablon részei (pl. címszöveg). A `content` alkönyvtárban pedig a saját munkánkat helyezhetjük el. Itt érdemes az egyes fejezeteket külön TeX állományokba rakni.

A diplomatervsablon (a kari irányelvek szerint) az alábbi fő fejezetekből áll:

1. 1 oldalas *tájékoztató* a szakdolgozat/diplomaterv szerkezetéről (`include/guideline.tex`), ami a végső dolgozatból törlendő,
2. *feladatkiírás* (`include/project.tex`), a dolgozat nyomtatott verzójában ennek a helyére kerül a tanszék által kiadott, a tanszékvezető által aláírt feladatkiírás, a dolgozat elektronikus verziójába pedig a feladatkiírás egyáltalán ne kerüljön bele, azt külön tölti fel a tanszék a diplomaterv-honlapra,
3. *címszöveg* (`include/titlepage.tex`),
4. *tartalomjegyzék* (`thesis.tex`),
5. a diplomatervező *nyilatkozata* az önálló munkáról (`include/declaration.tex`),
6. 1-2 oldalas tartalmi *összefoglaló* magyarul és angolul, illetve elkészíthető még további nyelven is (`content/abstract.tex`),
7. *bevezetés*: a feladat értelmezése, a tervezés célja, a feladat indokoltsága, a diplomatervezésnek rövid összefoglalása (`content/introduction.tex`),
8. sorszámmal ellátott *fejezetek*: a feladatkiírás pontosítása és részletes elemzése, előzmények (irodalomkutatás, hasonló alkotások), az ezekből levonható következtetések, a tervezés részletes leírása, a döntési lehetőségek értékelése és a választott megoldások indoklása, a megtervezett műszaki alkotás értékelése, kritikai elemzése, továbbfejlesztési lehetőségek,
9. esetleges *köszönetnyilvánítások* (`content/acknowledgement.tex`),
10. részletes és pontos *irodalomjegyzék* (ez a sablon esetében automatikusan generálódik a `thesis.tex` fájlban elhelyezett `\bibliography` utasítás hatására, a section 12.5ban leírtak szerint),
11. *függelékek* (`content/appendices.tex`).

A sablonban a fejezetek a `thesis.tex` fájlba vannak beillesztve `\include` utasítások segítségével. Lehetőség van arra, hogy csak az éppen szerkesztés alatt álló `.tex` fájlt fordítsuk le, ezzel lerövidítve a fordítási folyamatot. Ezt a lehetőséget az alábbi kód részlet biztosítja a `thesis.tex` fájlban.

```
\includeonly{  
    guideline,%  
    project,%  
    titlepage,%  
    declaration,%}
```

```
abstract,%
introduction,%
chapter1,%
chapter2,%
chapter3,%
acknowledgement,%
appendices,%
}
```

Ha az alábbi kódrészletben az egyes sorokat a % szimbólummal kikommentezzük, akkor a megfelelő .tex fájl nem fordul le. Az oldalszámok és a tartalomjegyék természetesen csak akkor billennek helyre, ha a teljes dokumentumot lefordítjuk.

12.7 Alapadatok megadása

A diplomaterv alapadatait (cím, szerző, konzulens, konzulens titulusa) a `thesis.tex` fájlban lehet megadni.

12.8 Új fejezet írása

A főfejezetek külön content könyvtárban foglalnak helyet. A sablonhoz 3 fejezet készült. További főfejezeteket úgy hozhatunk létre, ha új TeX fájlt készítünk a fejezet számára, és a `thesis.tex` fájlban, a `\include` és `\includeonly` utasítások argumentumába felvesszük az új `.tex` fájl nevét.

12.9 Definíciók, tételek, példák

Definition 1 (Fluxuskondenzátor térerőssége). Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Example 1. Példa egy példára. *Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.*

Theorem 1 (Kovács tétele). *Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.*

Bibliography

- [1] J3016B: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles - SAE International. URL https://www.sae.org/standards/content/j3016_201806/.
- [2] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.
- [3] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019.

Appendix